Control-M Workload Automation 8.0.00.300 Utilities Guide

March 2014

www.bmc.com

Contacting BMC Software

You can access the BMC Software website at http://www.bmc.com. From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

United States and Canada

Address	**BMC SOFTWARE INC**	**Telephone**	▪ **713 918 8800**	**Fax**	**713 918 8000**
	2101 CITYWEST BLVD		▪ **800 841 2031**		
	HOUSTON TX **77042-2827**				
	USA				

Outside United States and Canada

Telephone	**(01) 713 918 8800**	**Fax**	**(01) 713 918 8000**

BMC Software considers information included in this documentation to be proprietary and confidential. Your use of this information is subject to the terms and conditions of the applicable End User License Agreement for the product and the proprietary and restricted rights notices included in this documentation.

Restricted rights legend

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED -- RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2101 CITYWEST BLVD, HOUSTON TX 77042-2827, USA. Any contract notices should be sent to this address.

Customer support

You can obtain technical support by using the BMC Software Customer Support website or by contacting Customer Support by telephone or e-mail. To expedite your inquiry, see “Before contacting BMC.”

Support website

You can obtain technical support from BMC 24 hours a day, 7 days a week at http://www.bmc.com/support. From this website, you can:

- Read overviews about support services and programs that BMC offers
- Find the most current information about BMC products
- Search a database for issues similar to yours and possible solutions
- Order or download product documentation
- Download products and maintenance
- Report an issue or ask a question
- Subscribe to receive proactive e-mail alerts when new product notices are released
- Find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Support by telephone or e-mail

In the United States and Canada, if you need technical support and do not have access to the web, call 800 537 1813 or send an e-mail message to customer_support@bmc.com. (In the subject line, enter **SupID:<yourSupportContractID>**, such as SupID:12345). Outside the United States and Canada, contact your local support center for assistance.

Before contacting BMC

Have the following information available so that Customer Support can begin working on your issue immediately:

- Product information
  - Product name
  - Product version (release number)
  - License number and password (trial or permanent)
- Operating system and environment information
  - Machine type
  - Operating system type, version, and service pack or other maintenance level such as PUT or PTF
  - System hardware configuration
  - Serial numbers
  - Related software (database, application, and communication) including type, version, and service pack or maintenance level
- Sequence of events leading to the issue
- Commands and options that you used
- Messages received (and the time and date that you received them)
  - Product error messages

- Messages from the operating system, such as `file system full`
- Messages from related software

License key and password information

If you have questions about your license key or password, contact BMC as follows:

- (USA or Canada) Contact the Order Services Password Team at 800 841 2031, or send an e-mail message to ContractsPasswordAdministration@bmc.com.
- (Europe, the Middle East, and Africa) Fax your questions to EMEA Contracts Administration at +31 20 354 8702, or send an e-mail message to password@bmc.com.
- (Asia-Pacific) Contact your BMC sales representative or your local BMC office.

For the provisions described in the BMC License Agreement and Order related to third party products or technologies included in the BMC Product, see the "Third Party" subdirectory in the installation directory of this product.

Contents

Introduction to Utilities

Utilities can be used to perform many common Control-M tasks from the command prompt of any computer where Control-M components are installed.

Although many of the tasks performed by these utilities can be performed using Control-M Workload Automation or the Control-M Configuration Manager, the utilities enable you to work at computers that do not have the GUI or the Control-M Configuration Manager installed on them. By including a utility command in the command line of a job processing definition, you can automatically run the utility at a predetermined time or under a predetermined set of conditions.

You can perform the following:

Task	Description
Types of Utilities (on page 14)	Describes the different types of utilities and the to components they belong to.
Methods for running utilities (on page 15)	Describes the methods for running a utility, such as interactive, silent, and etc.
Considerations for running utilities (on page 18)	Describes important considerations when running a utility for Control-M/EM, Control-M/Server, and Control-M/Agent.
Utility reference table (on page 20)	Lists the utilities with an overview of some of the utility requirements.
Running Control-M/Server utilities as other users (on page 25)	Describes how to run Control-M/Server utilities as users other than administrators.
Running Control-M/Agent utilities as other users (on page 31)	Describes how to run Control-M/Agent utilities as users other than administrators.
Parameter name cross references (on page 34)	Lists the utility parameters and their equivalent Control-M/EM name.
Unsupported utilities (on page 628)	Lists the utilities that are not supported by BMC.

Abbreviations and conventions

Component	Convention
Control-M/Agent	Text and examples are given according to UNIX usage, unless otherwise stated. The default home directory of the UNIX user account under which Control-M/Agent is installed is <*agentHome*>.
Control-M/EM	Text and examples are given according to UNIX usage, unless otherwise stated. The default full path name of the home directory in which Control-M/EM is installed is **ctm_em,** *homeDirectory*.
Control-M/Server	Text and examples are given according to UNIX usage, unless otherwise stated. The default full path name of the home directory of the UNIX user account under which Control-M/Server is installed is <*$HOME*>**/ctm_server**, for example, <*$HOME*>**/ ctm_server/data**.

NOTE: Unless specifically stated, utilities that can run on UNIX can also be run on Linux and on Windows.

The following abbreviations are used:

Abbreviation	Description
Control-M/Agent	Control-M/Agent for UNIX and Microsoft Windows
Control-M/EM	Control-M/Enterprise Manager
Control-M/Server	Control-M/Server for UNIX and Microsoft Windows
Net	Control-M/EM Network (meaning, a set of jobs, related by prerequisite conditions, in the active database.)
<*$HOME*>, <*homeDirectory*>	Directory in which Control-M/EM is installed

The following conventions are used:

Convention	Description
key	When describing keystrokes, the name of a key (for example, **F1**) is in boldface type. When two keys are joined with "**+**" as in **Shift+F1**, hold down **Shift** while pressing **F1**.
Menu => Option	This represents an option selection sequence. `Users and Groups => Groups => Add` means that you first select **Users and Groups** from the menu bar. Select the **Groups** option from the submenu. Finally, select the **Add** option from the **Groups** submenu.

{ } (braces)	Braces indicate that at least one of the enclosed elements is required. `{<fileName> | <deviceName>| <mediaType>}` means that you must specify one of the variables.
{Option A|Option B}	The vertical bar is used to separate choices. For example, {AND|OR}, means that you specify either AND or OR.
[Option]	Square brackets are used to enclose parameters that are optional.
Code Samples	Format syntax, operating system terms, examples, and JCL scripts are presented in this typeface.
Italics	*Italic* type is used to emphasize important, and the first occurrence of new, terms. The titles of BMC Software component documentation are also displayed in italics.
Ellipsis	An ellipsis (. . .) indicates that you can repeat the preceding item or items as many times as necessary.
<variable>	In commands and parameters, angle brackets are used to enclose variable information. `cd <controlm_run_as>` In this command, you specify **cd** followed by the installation path of Control-M.
italic	An italic font is used for the name of publications.
Messages	Messages are presented in this font.
Wildcards or Mask Characters	Certain Control-M utilities and parameters support wildcards. These are also sometimes referred to as *mask characters*. A *mask* is a string value containing wildcard characters. The following wildcard characters are supported: ▪ **?** indicates any one character ▪ ***** indicates any number of characters Values containing wildcard characters must be enclosed in single or double quotation marks.

The following table lists the new and previous parameter names:

Parameter	Previously Known As
Host/Host Group	Node/Node group

Parameter	Previously Known As
Run as	Owner
Override Path	Override library
Variable	AutoEdit
Sub Application	Group
Created by	Author
Folder	Table
Order Method	User daily
Output	Sysout

Command line conventions

Parameters for running utilities from the command line can be specified in:

- Short format, for example, **-u** *user* **-p** *pass* **-s** *guiServerName*
- Long format, for example,
  -USERNAME *user*, **-PASSWORD** *pass*, **-HOST** *guiServerName*
- Uppercase or lowercase letters

The following table lists the parameters that can be specified in both short and long formats:

Parameter	Short Format	Long Format
User name	-u (or -U)	-username (or -USERNAME)
User password	-p (or -P)	-password (or -PASSWORD)
Host computer	-s (or -S)	-host (or -HOST)
Source file	-src (or -SRC)	-src_file (or -SRC_FILE)
Argument file	-arg (or -ARG)	-arg_file (or -ARG_FILE)
Output file	-out (or -OUT)	-out_file (or -OUT_FILE)

Most of the examples in this book use parameters in lowercase, and are specified in the short format.

NOTE: For Control-M/EM CLI utilities, write the user name in inverted commas (" "), if the username contains a space character.

Types of Utilities

This book describes utilities that can be run from computers on which the following components reside:

- Control-M/EM
- Control-M/Server
- Control-M/Agent

Utility Type	Description
Definition, ordering and monitoring	You can create, modify, delete, order and monitor job processing definitions with the utilities described in Definition, ordering, and monitoring (on page 38).
Folder and calendar	You can create and modify calendar definitions, folders, and SMART Folders and Sub Folders with the utilities described in Folders and Calendars (on page 244).
Command line interface	The Command Line Interface (cli) is a batch utility that enables you to perform the following operations (services) from the command line: ▪ Upload or download folders ▪ Order or force folders ▪ Order or force jobs ▪ Force jobs in a SMART Folder ▪ Upload or download calendars ▪ Delete job processing definitions from folders For information about cli, see Uploading, ordering and scheduling jobs and folders using the cli utility (on page 303).
New day procedure and user dailies	You can create and modify how the job ordering processing is automated using the utilities in New day procedure and order methods Control-M/Server utilities (on page 315).
Active database	You can monitor and impact the jobs running in the active database by running the utilities described in Affecting the active database Control-M/Server utilities (on page 326).
Communication, startup and troubleshooting	You can start up, shut down, and troubleshoot various Control-M components by running the utilities described in Communication, startup, and troubleshooting (on page 345).
Administration	You can monitor and manage selected elements of Control-M using the utilities, as described in Administration and configuration (on page 407).

Utility Type	Description
Database maintenance	You can maintain the Control-M/EM database using the utilities described in Database maintenance (on page 479).
Security	You can implement various levels of security using the utilities described in Security (on page 564).
Gather statistics and run reports	Generate statistics and reports using the utilities described in Statistics and reporting (on page 594).

Methods for running utilities

Utilities can be run using the following methods:

- Interactively running utilities from menus or from the Control-M Configuration Manager (on page 15)
- Automating utilities using Control-M jobs (on page 15)
- From the command line (on page 16)

Agentless job submission is supported, that is, submission and execution of jobs on computers that do not have a resident Control-M/Agent installed. These agentless computers are referred to as remote hosts. (Previously, a Control-M/Agent had to be installed on each computer where jobs would run.)

Interactively running utilities from menus or from the Control-M Configuration Manager

Many utilities can be run interactively using menus and/or the Control-M Configuration Manager.

Automating utilities using Control-M jobs

Many of the tasks performed by utilities can also be performed in real-time using the Job Properties and Folder Properties panes in Control-M for Databases and Control-M Workload Automation. However, by including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

➢ To automate utilities using Control-M:

1. In the Job Properties pane, do the following:
   a. Specify "command" as the **What** field.
   b. Specify the command line format of the utility in the **Command** field.
   c. Set up the automation using the fields in the **Scheduling** and **Condition** tabs.

2. Check in, and order the job.

The New Day Procedure, on a daily basis, will evaluate whether the utility should run based on the criteria you set up.

From the command line

Almost all Control-M utilities are specified using the command line. For more information about conventions for running utilities from the command line, see Command line conventions (on page 13).

If a command invoked from the agent computer contains embedded spaces, add **\"** after the first quote at the beginning of the command, and add **\"** at the end of the command (but prior to any parameters).

NOTE: "\"d:\program files\bmc software\control-m agent\exe_sleep\" 200"

Syntax on UNIX

Some Control-M/Agent utilities require special formatting for transmission to the Server computer. See Special Utility Parameter Formats below.

Values for utility parameters must not contain the apostrophe or single quote character.

When invoking these utilities from Control-M/Agent, the presence of a special character in the argument values may cause problems. The following command contains a back-slash before the string DELETEME:

```
ctmcreate -what COMMAND -jobname servertest -cmdline "ctmvar -action set -var
'%%#\DELETEME' -varexpr to_be_deleted"
```

When this command is invoked from Control-M/Agent, the back-slash before DELETEME may be "eaten" by the shell. To avoid this problem, add a back-slash before the special character that causes the problem (in this case, the original back-slash).

When invoking the ctmcreate utility or ctmdefine utility from Control-M/Agent with a date_ref of **$$$$**, put a back-slash before each $ as shown here:

```
ctmcreate -what command -cmdline ls -incond a '\$\$\$\$' AND ...
```

Special utility parameter formats

Commands invoked from UNIX agent computers are embedded in double quotes when sent to the Server computer. Therefore, use single quotes for command elements that must be within quotation marks. For example:

```
ctmcreate ...-cmdline "ls -l ’$HOME’"
```

For a description of syntax rules when invoking the ctmcreate utility, see ctmcreate (on page 168).

Using an input file (-input_file parameter)

Using an input file enables you to:

- prepare and save files of utility parameters that can be reused
- specify utility input longer than the number of characters allowed in the command line (10,000)

To specify parameters whose characters exceed 10,000 characters, or to save commands that you use often, specify the command in an input file and reference this file when you run the command for the utility using the -input_file parameter. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line.

For example, assume that you have to specify the ctmcreate utility. In this case the parameters with their values exceed the 10,000 character limit. You can specify the parameters and values in an input file, as follows:

```
ctmcreate -input_file <fileName>
```

The <*fileName*> variable is the name of a file that contains the ctmcreate parameters. For example:

-what command

-cmdline ls

Directing output from utilities

Certain Control-M/Server and Control-M/Agent utilities generate reports that can be directed to a file. Each such utility is identified by the inclusion of *<output>* among the utility's parameters.

- If output parameters are specified, output is routed to the specified file (for example, a file on the Control-M/Server computer).

```
ctmordck SYSTEM SYSTEM ~<controlmOwner>/ctm_server/user_a/udlist.txt
```

- If output parameters are not specified, the output is routed to the default output device (for example, the logical name of a disk).

When directing output to a file, do one of the following:

- Specify the full path name of the file (for example, `~<controlmOwner>/ctm_server/user1/rprt.txt).`
- Specify the relative name of the file to be placed in the **<***controlmUserDir***>/** directory.
- Redirect to the Control-M/Agent computer by specifying a full path name of the file after the redirection (**>**) character.

Considerations for running utilities

When running utilities on the various Control-M components:

- Considerations for running Control-M/EM utilities
- Considerations for running Control-M/Server utilities
- Considerations for running Control-M/Agent utilities

Considerations for running Control-M/EM utilities

When running Control-M/EM utilities, consider the following:

- Unless stated, utilities that can run on UNIX can also be run on Linux and on Windows.
- There are Control-M/EM database utilities that are implemented by using input and argument files written in Extensible Markup Language (XML). Instructions for preparing these XML files are presented in XML file preparation (on page 616).
- A valid Control-M/EM database user name and password are required to run Control-M/EM utilities. Additionally, you must have the appropriate authorization to either copy from or modify entities in Control-M/EM database. For more information, see Control-M/EM Authorizations.

Considerations for running Control-M/Server utilities

Users who are not Control-M administrators can run Control-M/Server utilities by following the instructions provided under Running Control-M/Agent utilities as other users (on page 31).

Considerations for running Control-M/Agent utilities

Agent utilities that run on a Windows computer by a user other than the administrator may not have access to certain data or functions. Agent utilities that run on a UNIX computer can be run by users other than the agent account owner, after the user has loaded the agent account environment. For more information, see Running Control-M/Agent utilities as other users (on page 31).

All Control-M/Agent utilities support the -agent <agent name> parameter. The variable <agent name> represents the name of the Control-M/Agent specified during the installation procedure.

Where multiple Control-M/Agents reside on a computer, the -agent parameter determines which Control-M/Agent will manage the utility. If the ctmagcfg or ag_diag_comm utilities are run without specifying the -agent parameter, the user is prompted to select the Control-M/Agent. For all other utilities, if the -agent parameter is not specified, the default Control-M/Agent is used.

EXAMPLE: Assume a computer has two Agents, **Default** and **Saturn**. To add a user to **Default**, use the following command:

```
ctmpwd -action add -user user2 -password 123456 -agent Default
```

-or-

```
ctmpwd -action add -user user2 -password 123456
```

To add a user to **Saturn**, use the following command:

```
ctmpwd -action add -user saturn_user2 -password 123456 -agent Saturn
```

Utility reference table

Use the following table as a handy reference as to what can be specified with the utility you are working with. You can determine, for example, if variables can be specified in the utility or if the utility supports the submission of parameter information in an input file.

Utility	Requires Control-M active?	SQL Access	Only Administrator can run utility?	Can be run as a batch job?	Supports Variables?	Creates reports?	Supports input file?	Can be run interactively?	Can be run from agent computer	Can be run from a remote host computer
_exit									Yes	
_sleep									Yes	
ag_diag_comm									Yes	
ag_ping									Yes	
ctmagcfg									Yes	
ctmunixcfg									Yes	
ctmwincfg									Yes	
set_agent_mode									Yes	
ctm_agstat	Yes		Yes							
ctm_backup_bcp		Yes	Yes	Yes						
ctm_diag_comm	Yes	Yes	Yes	Yes						
ctm_menu	Yes	Yes	Yes	No	Yes			Yes		
ctm_restore_bcp		Yes	Yes	Yes						
ctmagcln	Yes	Yes	Yes	Yes						
ctmcalc_date		Yes		Yes						
ctmcontb	Yes	Yes		Yes	Yes				Yes	Yes
ctmcreate	Yes	Yes		Yes	Yes				Yes	Yes

Utility	Requires Control-M active?	SQL Access	Only Admini-strator can run utility?	Can be run as a batch job?	Supports Variables?	Creates reports?	Supports input file?	Can be run interactively?	Can be run from agent computer	Can be run from a remote host computer
ctmdefine		Yes		Yes	Yes				Yes	
ctmdeffolder		Yes		Yes	Yes				Yes	Yes
ctmdefsubfolder		Yes		Yes	Yes				Yes	Yes
ctmdiskspace				Yes						
ctmexdef		Yes		Yes						
ctmfw				Yes					Yes	
ctmgetcm	Yes	Yes		Yes						
ctmhostmap		Yes	Yes	Yes						
ctmipc	Yes			Yes						
ctmldnrs		Yes		Yes		Yes	Yes			
ctmloadset	Yes	Yes		Yes					Yes	Yes
ctmkeygen		Yes		Yes						
ctmkeystore_ mng			Yes							
ctmkilljob	Yes	Yes		Yes					Yes	Yes
ctmhostgrp		Yes		Yes						Yes
ctmorder	Yes	Yes		Yes					Yes	Yes
ctmordck		Yes		Yes						
ctmping	Yes	Yes		Yes						
ctmpsm		Yes								Yes
ctmrpln		Yes		Yes						

Utility	Requires Control-M active?	SQL Access	Only Administrator can run utility?	Can be run as a batch job?	Supports Variables?	Creates reports?	Supports input file?	Can be run interactively?	Can be run from agent computer	Can be run from a remote host computer
ctmruninf										
ctmshtb		Yes	Yes	Yes						
ctmshout	Yes	Yes		Yes						Yes
ctmspdiag		Yes	Yes							Yes
ctmstvar		Yes		Yes	Yes					
ctmsuspend	Yes	Yes	Yes							
ctmsweep		Yes		Yes						
ctmudchk		Yes		Yes						
ctmudlst		Yes	Yes	Yes						
ctmudly	Yes	Yes	Yes	Yes					Yes	Yes
ctmvar		Yes		Yes	Yes				Yes	Yes
ctmwhy		Yes		Yes						
db_check_space		Yes	Yes	Yes				Yes		
dbversion		Yes		Yes						
shctm			Yes	Yes						
show_ca			Yes	Yes						
shut_ca			Yes	Yes						
shut_ctm			Yes	Yes						
shutdb		Yes								
start_ca			Yes	Yes						

Utility	Requires Control-M active?	SQL Access	Only Admini - strator can run utility ?	Can be run as a batch job?	Supports Variables?	Creates reports?	Supports input file?	Can be run interactively?	Can be run from agent computer	Can be run from a remote host computer
start_ctm			Yes	Yes						
startdb		Yes								
bim_report		Yes	Yes	Yes		Yes				
build_db		Yes						Yes		
ccmcli				Yes			Yes			
cli	Yes			Yes						
copydefcal				Yes			Yes			
copydefjob				Yes			Yes			
ctl		Yes	Yes	Yes			Yes			
db_check		Yes	Yes	Yes				Yes		
defcal				Yes			Yes			
defjob				Yes			Yes			
deffolder				Yes			Yes			
deldefjob				Yes			Yes			
duplicatedefjob				Yes			Yes			
ecactltb	Yes	Yes		Yes		Yes			Yes	Yes
ecaqrtab	Yes	Yes		Yes		Yes	Yes		Yes	Yes
em_data_ collector		Yes	Yes	Yes		Yes		Yes		
em_database_ menu		Yes	Yes					Yes		

Utility	Requires Control-M active?	SQL Access	Only Admini-strator can run utility?	Can be run as a batch job?	Supports Variables?	Creates reports?	Supports input file?	Can be run interactively?	Can be run from agent computer	Can be run from a remote host computer
em_rebuild_database		Yes	Yes				Yes	Yes		
em_SQL		Yes	Yes					Yes		
emcryptocli				Yes			Yes			
migrate_dc		Yes	Yes	Yes					Yes	
emreportcli		Yes		Yes		Yes	Yes			
erase_audit_data		Yes	Yes (database owner)					Yes		
exportdefcal				Yes			Yes			
exportdefjob				Yes			Yes			
exportdeffolder				Yes			Yes			
loader		Yes	Yes	Yes			Yes			
loopdetecttool				Yes		Yes	Yes			
orbadmin				Yes						
purge_runinfo		Yes	Yes	Yes				Yes		
purge_xalerts		Yes	Yes	Yes				Yes		
sweep		Yes	Yes	Yes			Yes	Yes		
updatedef				Yes			Yes			
util		Yes	Yes	Yes			Yes	Yes		

Running Control-M/Server utilities as other users

Enables users who are not Control-M administrators to run Control-M/Server utilities.

To allow a user other than the Control-M administrator to run Control-M/Server utilities on a Windows computer, change the permissions of the following directories:

- temp
- proclog
- prflag

The user requires the permission to write to these directories.

To access Control-M/Server utilities from a UNIX user other than the Control-M/Server account owner, the following modifications must be made to the user environment:

- Define variables in the user environment
- Add an executable library to the user's path
- Assign Read/Write permissions

The following environment variables must be defined:

- CONTROLM_SERVER
- CONTROLM_USER
- CONTROLM_DATABASE
- CONTROLM_MIRROR_USER
- CONTROLM_MIRROR_DATABASE
- CTM_DATABASE_TYPE
- MIRROR_DB_SERVER
- machine
- LIBPATH, or LD_LIBRARY_PATH, or SHLIB_PATH

The following table lists additional environment variables that are required, according to the type of database:

For SYBASE	For ORACLE	For PostgreSQL
SYBASE	ORACLE_BASE	DB_TYPE
DSLISTEN	ORACLE_HOME	PGHOME
DSQUERY	ORACLE_SID	PGDATA
DSCONSOLE	NLS_LANG	PGPORT

For SYBASE	For ORACLE	For PostgreSQL
MIRROR_DSLISTEN		PGHOST
MIRROR_DSQUERY		PGDATABASE
SYBASE_OCS, with the value of the Sybase client version (OCS-12_5 or OCS-15_0)		PGUSER
		PGSYSCONFDIR
		PGSERVICE
		LD_LIBRARY_PATH/LIBPATH/ SHLIBPATH
		DBU_BIN
		MIRROR_DSQUERY

Specify the following command for each variable from the Control-M user:

- `echo $<variableName>`
- `echo $SYBASE`

Use the formats in the following table to define the shared library path variables for each database in the user environment, depending on the server computer type:

Computer	Shared library environment variable name
AIX	LIBPATH
HP-UX	SHLIB_PATH
Solaris	LD_LIBRARY_PATH
Linux x86_64	LD_LIBRARY_PATH

For Sybase

- Set the shared library environment variable value to:

  ```
  ~<controlmOwner>/ctm_server/exe_<opSysID>:$SYBASE/$SYBASE_OCS/lib
  ```

  The variables in the library path are:

  - `<opSysID>` is the identifier for the type of operating system
  - `<controlmOwner>` is the user account home

    $SYBASE is the value for SYBASE from

$SYBASE_OCS is the value for SYBASE from

For Oracle

- Set the Shared library environment variable value to:

  ```
  ~<controlmOwner>/ctm_server/oracle/lib:~<controlmOwner>/ctm_server/
  exe_<opSysID>
  ```

  The variables in the library path are:

  - `<opSysID>` is the identifier for the type of operating system from
  - `<controlmOwner>` is the user account home from

For PostgreSQL

- Set the Shared library environment variable value to:

  ```
  ~<controlmOwner>/ctm_server/exe_<opSysID>:~<controlmOwner>/pgsql/lib
  ```

  The variables in the library path are:

  - **<*opSysID*>** is the identifier for the type of operating system from
  - **<*controlmOwner*>** is the user account home from

The following table describes the shared library path variables for Sybase, Oracle, and PostgreSQL:

Variable	Description
<*Control-M/ServerPath*>	The default full path name of the Windows home directory under which Control-M/Server is installed.
~<*controlmOwner*>	The default full path name of the UNIX user account home directory under which Control-M/Server is installed.
<*opSysID*>	Identifier for the type of operating system on the server computer (AIX, HP-UX-11, HP-UX-ia64, Solaris, Linux).

Setting environmental variables

- Using csh or tcsh

  Specify the following command for each variable:

  ```
  setenv <variable-name> <value>
  ```

Sybase

```
setenv CONTROLM_SERVER /home/controlm/ctm_server

setenv SYBASE /home/controlm/sybase
setenv DSLISTEN CTRLM
setenv DSQUERY CTRLM
setenv DSCONSOLE CTRLM
setenv MIRROR_DSQUERY MIRROR
setenv MIRROR_DSLISTEN MIRROR
setenv ControlM_USER ctrlm
setenv ControlM_DATABASE ctrlm
```

```
setenv MIRROR_DB_SERVER ctrlm_mirror
setenv CTM_DATABASE_TYPE SYBASE
setenv CONTROLM_MIRROR_USER mirror_ctm
setenv CONTROLM_MIRROR_DATABASE mirror_ctm
setenv machine ${ControlM_SERVER}/scripts/sharch
setenv LD_LIBRARY_PATH ~<controlm_owner>/ctm_server/exe_Solaris:
/home/controlm/sybase/OCS-12_5/lib
```

Oracle

```
setenv CONTROLM_SERVER /home/ora_ctm/ctm_server
setenv ORACLE_HOME /home/ora_ctm/ctm_server/oracle
setenv ORACLE_SID ctrlm
setenv ORACLE_BASE /home/ora_ctm/ctm_server/oracle
setenv NLS_LANG AMERICAN_AMERICA.WE8ISO8859P1
setenv CONTROLM_USER ctrlm
setenv CONTROLM_DATABASE ctrlm
setenv MIRROR_DB_SERVER ctrlm_mirror
setenv CTM_DATABASE_TYPE ORACLE
setenv CONTROLM_MIRROR_USER mirror_ctm
setenv CONTROLM_MIRROR_DATABASE mirror_ctm
setenv machine ${CONTROLM_SERVER}/scripts/sharch
setenv LD_LIBRARY_PATH /home/ora_ctm/ctm_server/exe_Linux-x86_64:
/home/ora_ctm/ctm_server/oracle/lib
```

PostgreSQL

```
setenv CONTROLM_SERVER /home1/ctm640pg/ctm_server
setenv CTM_DATABASE_TYPE PGSQL
setenv CONTROLM_DATABASE ctrlm640
setenv MIRROR_DSQUERY mirror
setenv CONTROLM_MIRROR_USER mirror2
setenv CONTROLM_MIRROR_DATABASE mirror2
setenv PRIMARY_DB_SERVER CTRLM
setenv MIRROR_DB_SERVER mirror
setenv BU_BIN /home1/ctm640pg/ctm_server/exe_Solaris
setenv PGPORT 5432
setenv PGDATABASE ctrlm640
setenv PGUSER ctmuser
setenv PGHOME /home1/ctm640pg/pgsql
setenv PGDATA /home1/ctm640pg/pgsql/data

setenv CONTROLM_USER ctmuser

setenv machine Solaris

setenv PG_HOST ctrlm640

setenv PG_SERVICE db

setenv LD_LIBRARY_PATH
/home1/ctm640pg/ctm_server/exe_Solaris:/home1/ctm640pg/pgsql/lib

setenv
:/home1/ctm640pg/pgsql/lib32:/home1/ctm640pg/ctm_server/exe_Solari
s:/home1/ctm640pg/pgsql/lib:/home1/ctm640pg/pgsql/lib32:/home1/ctm
640pg/ctm_agent/ctm/exe:/home1/ctm640pg/ctm_agent/ctm/exe

setenv DB_TYPE SERVER

setenv PGSYSCONFDIR /home1/ctm640pg/pgsql/etc
```

- Using other shells

  When using other shells (for example, sh, ksh), specify the following command for each variable:

  ```
  <variable-name>=<value>; export <variable-name>
  ```

Sybase

```
CONTROLM_SERVER=/home/controlm/ctm_server; export CONTROLM_SERVER

SYBASE=/home/controlm/sybase; export SYBASE
DSLISTEN=CTRLM; export DSLISTEN
DSQUERY=CTRLM; export DSQUERY
DSCONSOLE=ctrlm; export DSCONSOLE
MIRROR_DSQUERY=MIRROR; export MIRROR_DSQUERY
MIRROR_DSLISTEN=MIRROR; export MIRROR_DSLISTEN
CONTROLM_USER=ctrlm; export CONTROLM_USER
CONTROLM_DATABASE=ctrlm; export CONTROLM_DATABASE

LIBPATH=/home/controlm/ctm_server/exe_HP-UX-11:/home/controlm/syba
se/OCS-12_5/lib; export LIBPATH
```

Oracle

```
CONTROLM_SERVER=/home/controlm/ctm_server; export CONTROLM_SERVER

CONTROLM_USER=ctrlm; export CONTROLM_USER
CONTROLM_DATABASE=ctrlm; export CONTROLM_DATABASE

LIBPATH=/home/controlm/ctm_server/exe_HP-UX-ia64:
/home/controlm/oracle/lib; export LIBPATH
```

PostgreSQL

```
CONTROLM_SERVER=/home1/ctm640pg/ctm_server

CTM_DATABASE_TYPE=PGSQL

CONTROLM_DATABASE=ctrlm640

MIRROR_DSQUERY=mirror
```

```
CONTROLM_MIRROR_USER=mirror2
CONTROLM_MIRROR_DATABASE=mirror2
PRIMARY_DB_SERVER=CTRLM
MIRROR_DB_SERVER=mirror
DBU_BIN=/home1/ctm640pg/ctm_server/exe_Solaris
PGPORT=5432
PGDATABASE=ctrlm640
PGUSER=ctmuser
PGHOME=/home1/ctm640pg/pgsql
PGDATA=/home1/ctm640pg/pgsql/data
CONTROLM_USER=ctmuser
machine=Solaris
PG_HOST=ctrlm640
PG_SERVICE=db
LD_LIBRARY_PATH=/home1/ctm640pg/ctm_server/exe_Solaris:/home1/ctm6
40pg/pgsql/lib
:/home1/ctm640pg/pgsql/lib32:/home1/ctm640pg/ctm_server/exe_Solari
s:/home1/ctm640pg/pgsql/lib:/home1/ctm640pg/pgsql/lib32:/home1/ctm
640pg/ctm_agent/ctm/exe:/home1/ctm640pg/ctm_agent/ctm/exe
DB_TYPE=SERVER
PGSYSCONFDIR=/home1/ctm640pg/pgsql/etc
```

Adding an executable library to the path of the user

The executable directory of Control-M for Databases must be added to the path of the user. This path must include:

```
~<controlm_owner>/ctm_server/scripts
```

- Using csh or tcsh

  Use the following command to modify the path when using csh or tcsh:

  - **Sybase:**

    ```
    set path=($path ${SYBASE}/OCS-12_5/bin
    ~<controlm_owner>/ctm_server/exe_<OS_ID>)
    ```

  - **Oracle**

    ```
    set path=($path ${ORACLE_HOME}/bin
    ~<controlm_owner>/ctm_server/exe_<OS_ID>)
    ```

```
set path=($path $ {ORACLE_HOME}/bin~<controlm_owner>/ctm_server/exe_Solaris)
```

  - **Using other shells**

    Use the following command to modify the path when using other shells (for example, sh, ksh):

```
      PATH="$PATH: ~<controlm_owner>/ctm_server/exe_<OS_ID>"
PATH="$PATH: ~<controlm_owner>/ctm_server/exe_HP-UX-ia64"
```

Assign Read/Write permissions

The user must have the following read/write permissions:

Permission type	Directory
Read permission for file	**<*controlm-directory*>/.controlm**
Read and Write permission for all files in directory	**<*controlm_owner*>/ctm_server/prflag**
If mirroring is enabled, Read permission for file	**<*controlm-directory*>/.controlm_mirror**
(*PGSQL only*) Write permissions for all files in directory	**<*controlm-directory*>/pgsql/etc/pg_service.conf**

You must provide security access to these files as follows:

- Give group controlm read permission for the files.
- All users who require access to Control-M utilities should belong to the same group as the Control-M/Server account as their primary group.

Running Control-M/Agent utilities as other users

(*Windows only*) If users, other than the administrator, invoke agent utilities from Control-M/Agent, the utilities will run. However, not all the features and data will be available.

(*UNIX only*) To enable users, other than the Control-M/Agent user, to invoke agent utilities from Control-M/Agent, add the following environment variables to **.cshrc** or **.profile**:

- **Add to .cshrc**

  ```
  set path = ( ${path} <agentHome>/ctm/scripts <agentHome>/ctm/exe )
  setenv CONTROLM "<agentHome>/ctm"
  if ( ${?LD_LIBRARY_PATH} ) then
  setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:<agentHome>/ctm/exe
  else
  setenv LD_LIBRARY_PATH "<agentHome>/ctm/exe"
  endif
  ```

- **Add to .profile**

  ```
  CONTROLM=<agentHome>/ctm
  export CONTROLM
  PATH=$PATH:<agentHome>/ctm/exe:<agentHome>/ctm/scripts
  ```

```
export PATH
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:<agentHome>/ctm/exe
export LD_LIBRARY_PATH
```

<*agentHome*> stands for the Control-M/Agent account home directory.

NOTE: Use this procedure for agent utilities other than agent configuration utilities. BMC recommends that only the Control-M/Agent user be enabled to change the agent configuration.

A "behind the scenes look" at Control-M utilities

The following are discussed in this section:

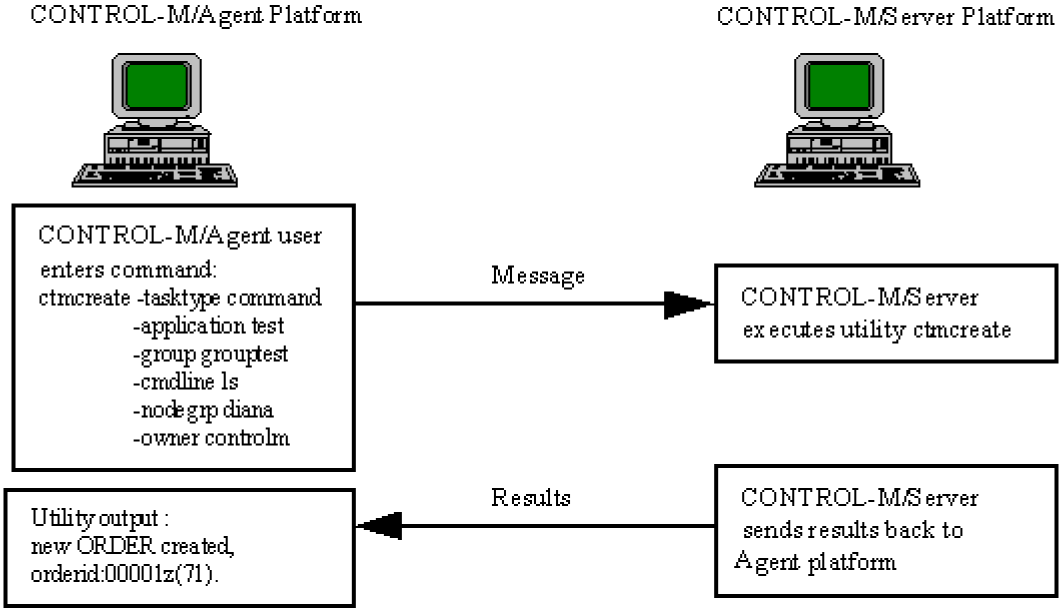
- Utility workflow between Control-M/Server and Control-M/Agent
- Optimizing how utilities run
- Optimizing Control-M/EM requests

Utility workflow between Control-M/Server and Control-M/Agent

Some of these utilities are executed by Control-M/Server. Their output is sent to the agent computer. The processing workflow is illustrated in the figure below.

Most utilities that create a job in the Control-M/Server Active Jobs database are interactive when invoked from the server computer, but not interactive when invoked from the agent computer. When invoked from the agent, they must be invoked with all the required parameters.

Figure 1: Control-M/Server utility workflow


CONTROL-M/Agent Platform
CONTROL-M/Server Platform
CONTROL-M/Agent user enters command:
Message
CONTROL-M/Server executes utility ctmcreate
Results
CONTROL-M/Server sends results back to Agent platform

```
ctmcreate -tasktype command
          -application test
          -group grouptest
          -cmdline ls
          -nodegrp diana
          -owner controlm
```

```
Utility output :
new ORDER created,
orderid:00001z(71).
```

NOTE: If the primary Control-M/Server does not respond to a Control-M/Agent request to execute a utility (other than ag_ping), the request is automatically redirected to the first non-primary Server listed in the Authorized Control-M/Server Hosts parameter. If the redirection is successful, that agent continues to work with the replacement server.

Optimizing how utilities run

Some Control-M/Server utilities can be invoked from Control-M/Agent using the same command line syntax. When such a utility is invoked by the agent, the agent sends the command line, by way of a message, to the server for execution.

Setting timeout intervals

The Agent-to-Server communication timeout intervals are described in Defining a Control-M/Agent component. If the agent requests a utility that runs on the server and there is no response within the timeout interval, the requested action will fail.

You can modify this timeout interval by using the Control-M/Agent System Parameters window in the Control-M Configuration Manager. However, increasing this timeout interval tends to reduce Control-M/Agent performance.

Optimizing Control-M/EM requests

You can set the Control-M/EM Gateway (GTW) parameters to optimize requests and communication between Control-M/EM and Control-M/Server. Parameters in the **Defaults.rsc** file that affect timeout values are discussed below. For more information, see Defining a Control-M/EM component.

Setting timeout values

The following modifications to timeout values can be made:

- You can modify the following parameters in the **Defaults.rsc** file to set timeout values:
  - download_timeout
  - upload_timeout
  - usreq_timeout (affects user requests like hold, free, kill, delete, and save)
  - inforeq_timeout (affects information requests timeout like output, documentation, and log)
  - keep_alive_timeout
  - sync_timeout
- You can optimize the SSL synchronization timeout by modifying the SSLSyncTime system parameter.

  For more information about modifying the **Defaults.rsc** file, see System configuration.
- Setting a timeout value for a GUI Server affects only requests that the GUI Server invokes as a CORBA client. Examples of such requests are:
  - updating the GUI with Viewpoint information
  - sending results of a callback action, such as Upload Folder or Holding an active job

  To prevent disconnections from slow clients, set the GSR AddJobsChunkSize system parameter to 100 (the default chunk size is 1000). BMC recommends decreasing the chunk size instead of increasing the timeout value.

Parameter name cross references

This table lists each parameter available for the utility and the name by which the parameter appears in the Job/Folder properties pane in Workload Automation and *Control-M Parameters*.

Utility parameters	Control-M/EM parameter
ADJUST_COND	Adjust condition
APPLFORM	Application Form
APPLICATION	Application
APPLTYPE	Application Type
APPLVER	Application Version
VARIABLE	Variable Assignments
CMDLINE	Command Line
CMVER	Application Plug-in Version
CONFIRM	Confirm Submission
Control	Control Resources
CRITICAL	Critical
CYCLIC	Cyclic
DESCRIPTION	Description
DO REMEDY	Notify - Remedy
DOVARIABLE	Set Variable
DOCLIB	Doc Library
DOCMEM	Doc Member
DOCOND	Add/Remove condition
DOFORCEJOB	Order Job
DOMAIL	Notify - Mail

Utility parameters	**Control-M/EM parameter**
DOOK	Do OK
DORERUN	Rerun Job
DOSHOUT	Notify
CAL_ANDOR	AND/OR
CONFCAL	Confirmation Calendar
CONFIRM	Confirm Submission
DATE	Dates
DATEFROM	Start Date
DATEUNTIL	End Date
DAYS	Days
DAYSCAL	Days Calendar
INTERVAL_SEQUENCE	Rerun using the following Interval Sequence
SPECIFIC_TIMES	Run at
TOLERANCE	Tolerance
EMBEDDED_SCRIPT	Embedded Scripts/JCL
RBC	Rule-Based Calendar
RELATIONSHIP	Relationship
RETRO	Retroactively order job that its scheduled date has passed
SHIFT	If day is not confirmed
WEEKCAL	Week days Calendar
WEEKDAYS	Days of week
TIMEUNTIL	Time
DOSTOPCYCLIC	Stop Cyclic Run

Utility parameters	Control-M/EM parameter
DOOUTPUT	Handle Output
SUB_APPLICATION	Sub application
INTERVAL	Interval
INCOND	In Conditions
INTERVALFROM	All intervals are from job's
JOBNAME	Job Name
MAXRERUN	Maximum reruns
MAXWAIT	Keep active
HOSTGRP	Host (/Group)
MEMLIB	Member Library
MEMNAME	File Name
ODATE	Order Date
ON	On Statement/Code
OUTCOND	Out Conditions
OVERRIDE_PATH	Override path
RUN_AS	Run as
PRIORITY	Priority
QUANTITATIVE	Quantitative Resources
SHOUT	Notification
OUTPUT	Output Handling
FOLDER	Folder
TASKTYPE	What
TIMEFROM	Time from

Utility parameters	Control-M/EM parameter
TIMEUNTIL	Time to

Chapter 2

Definition, ordering, and monitoring

The definition, ordering, and monitoring utilities are used to create and define job processing definitions:

NOTE: Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

The following table describes the utilities for various actions on job processing definitions.

<table>
<tr><th colspan="2">Utility Type</th><th>Description</th></tr>
<tr><td colspan="2">_exit (on page 242)</td><td>Create a job directly in the Active Jobs database.</td></tr>
<tr><td colspan="2">ctmdefine (on page 179)</td><td>Create a job directly in the Control-M/Server database.</td></tr>
<tr><td colspan="2">ctmexdef (on page 189)</td><td>Export jobs from the Control-M/Server database to an ASCII file for use with ctmcreate or ctmdefine.</td></tr>
<tr><td colspan="2">ctmfw File Watcher utility (on page 191)</td><td>Detect completion of file transfer activity.</td></tr>
<tr><td colspan="2">ctmimptb (on page 203)</td><td>Import SMART Folders that were exported with the exportdeffolder utility.</td></tr>
<tr><td colspan="2">ctmkilljob (on page 206)</td><td>Terminate a Control-M job and its associated processes.</td></tr>
<tr><td colspan="2">ctmorder (on page 208)</td><td>Order jobs from a SMART Folder in the Control-M/Server database.</td></tr>
<tr><td colspan="2">ctmpsm (on page 215)</td><td>Interactively perform functions on jobs or conditions directly in the Active database.</td></tr>
<tr><td colspan="2">ctmsweep (on page 236)</td><td>Delete non-active jobs from the Control-M/Server database.</td></tr>
<tr><td colspan="2">ctmwhy (on page 240)</td><td>Report why a job waiting in the Active Jobs database is not submitted.</td></tr>
<tr><td rowspan="2">emdef utility for jobs (on page 40)
(The emdef utility parameters are listed in the right</td><td>copydefjob (on page 61)</td><td>Create a job by copying an existing one.</td></tr>
<tr><td>defjob (on page 41)</td><td>Imports job processing definitions directly into Control-M/EM database</td></tr>
</table>

Utility Type		Description
column.)	deldefjob (on page 81)	Deletes specified job processing definitions Control-M/EM database.
	duplicatedefjob (on page 86)	Duplicates an existing job definition in the same data center and Folder.
	exportdefjob (on page 110)	Export jobs from the Control-M/EM database and save them in a file.
	loopdetecttool (on page 115)	Check jobs to avoid condition loops.
	updatedef (on page 122)	Update job processing, folders, SMART Folders and Sub-folders.

emdef utility for jobs

The emdef is a command line utility used to make various modifications to job definitions in the Control-M/EM database. The emdef uses the following parameters:

Utility Type	Description
defjob (on page 41)	The defjob parameter of the emdef utility imports job processing definitions directly into Control-M/EM database.
copydefjob (on page 61)	The copydefjob parameter of the emdef utility creates a new job definition in the Control-M/EM database that is similar to a specified existing definition.
deldefjob (on page 81)	The deldefjob parameter of the emdef utility deletes specified job processing definitions from a SMART Folder in the Control-M/EM database.
duplicatedefjob (on page 86)	The duplicatedefjob parameter of the emdef utility makes a copy of an existing job definition in the same data center and SMART Folder.
exportdefjob (on page 110)	The exportdefjob parameter of the emdef utility exports job processing definitions from a folder in the Control-M/EM database to an output file.
loopdetecttool (on page 115)	The loopdetecttool parameter of the emdef utility checks job processing definitions to determine if conditions are defined in a way that would cause loops.
updatedef (on page 122)	The updatedef parameter of the emdef utility updates (modifies) specified parameter values in the following definitions in the Control-M/EM database: ▪ Job processing definitions ▪ Folder definitions ▪ SMART Folder definitions ▪ Sub Folder definitions

defjob

The defjob parameter of the emdef utility imports job processing definitions directly into Control-M/EM database. To run the defjob utility, see Running the defjob utility (on page 41).

defjob reads job processing definitions from a plain text input file written in XML format, see defjob input file (on page 43).

Each job processing definition in the Control-M/EM database has a JOBISN field that contains a job ID number. JOBISN is the JOB_ID field in the def_job folder. Many jobs can have the same JOBISN number. However, the JOBISN number is unique in each Folder. Use the Control-M Workload Automation folder manager to determine the job ID. Double-click the required folder to get the job list, including the jobs IDs.

If a job that is being imported contains a JOBISN number that already exists in the Control-M/EM database, defjob overwrites the existing job processing definition with the new job processing definition. If a JOBISN value is not specified, defjob imports the job processing definition into the database as a new job.

Running the defjob utility

This procedure describes how to run the defjob utility, which enables you to import job processing definitions directly into Control-M/EM database.

➢ To run the defjob utility:

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*).

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the `<EM Home>\<instanceName>\bin` directory.

2. Enter either of the following commands:

-
  ```
  emdef defjob [-u <user> [-p <password>] | -pf <password file>]
  -s <GUI Server Name> -src <XML file name> [/a]
  ```

-
  ```
  emdef defjob [-USERNAME <user> [-PASSWORD <password>] | -PASSWORD_FILE
  <password file>] -HOST <GUI Server Name> -SRC_FILE <XML file name> [/a]
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the defjob parameters and switches, see defjob parameters (on page 42) and defjob switches (on page 42) .

The defjob input file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

defjob parameters

The following table lists the defjob parameters:

Parameter	Description
<user name>	Control-M/EM user name.
<password>	Control-M/EM user password.

<password file name>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName*>
	password=<*password*>

<GUI Server Name>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<XML file name>	The path and name of the XML file containing the defjob specifications. For more information, see Control-M Variable facility .

defjob switches

The following table lists the defjob switches:

Switch	Description
?	Displays utility's description and available options.
/a	Accept all. The /a switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The /a switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to 2 or 3.
/c	Operates on a chunk of jobs at a time to reduce process memory.
-cs<chunk size>	Set the number of jobs in a chunk.
/v	Used to receive verbose messages.

defjob input file

The job definitions that you create for use with the defjob utility are written in XML format and saved in a text file.

When this file is invoked, its contents are passed to the Control-M/EM databases database. Instructions for creating input files are in Control-M Variable facility .

The following rules apply to the input file of the defjob utility:

- More than one job can be specified in a defjob file.
- The input file is case-sensitive.
- Although many parameters are optional, certain parameters are required depending on the option specified for the TASKTYPE parameter. For more information, see What in *Control-M Parameters*.
- All parameter values (such as strings or digits) must be enclosed in quotation marks (for example, JOBNAME="Job1").
- Condition dates must be specified in mmdd format. Time must be specified in hhmm format.
- A parameter requiring more than one entry can be repeated as many times as necessary. For example, if a job must wait for several prerequisite conditions, specify a separate INCOND parameter for each prerequisite condition.
- Each ON_STMT or ON_STEP parameter must be followed by at least one DO parameter. DO parameters are dependent upon the last ON_STMT or ON_STEP parameter preceding them.

The values that follow can have more than one line of text:

- Do Mail Message
- Do Remedy Summary
- Do Remedy Description
- Instream JCL (for z/OS)

The multi line value is formatted in the following manner:

- A four digit number that indicates the length of the line
- Text following the number
- For example: "0011hello world0012just kidding"--

  If the UseMultiLineNewFormatDisplay system parameter is set to 1, then all the values appear with &#13;&#10; as line delimiters. If set to 0, or not set at all, then the following values appear as described in the above format.

XML is comprised of elements and attributes. Each element can contain attributes and sub-elements.

For more information on the input file parameters for the defjob utility, see defjob input file parameters (on page 44), and defjob input file example (on page 60).

defjob input file parameters

The following table lists the defjob input file parameters:

Parameter	**Description**
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the **.dtd** file.	
DEFJOB	Indicates the XML root element. Job processing definitions are placed between the opening and closing DEFJOB tags. One or more jobs can be specified. Each individual job is enclosed by JOB /JOB tags.
JOB	Indicates the opening and closing tags of a single job definition. Parameters of the job are listed between the tags.
DATACENTER	Name of the Control-M installation to which the job belongs. String. Mandatory.
PARENT_FOLDER	Name of the parent folder to which the job's folder belongs. String. Mandatory.
FOLDER_NAME	Name of the SMART Folder to which the job belongs. String. Mandatory.
FOLDER_DSN	(*z/OS only*) Name of the library that contains the SMART Folder. String. Mandatory.
JOBNAME	Name of the job processing definition. String. Mandatory. On a Windows computer, JOBNAME must comply with Microsoft naming conventions (for example, it cannot contain / and \ characters).
FILE_NAME	Name of the file that contains the job script. String. Optional.
SUB_APPLICATION	Name of the group to which the job belongs. Used for logical groupings of jobs. String. Mandatory.
APPLICATION	Name of the application to which the job's group belongs. Used as a descriptive name for related groups of jobs. String. Mandatory.

Parameter	Description
TASKTYPE	Type of job (task) to be performed by Control-M. Mandatory. Valid values: ▪ Job ▪ Detached ▪ Command ▪ Dummy ▪ External (*z/OS only*) Valid values: ▪ Job ▪ Started_Task
CREATED_BY	Control-M/EM user who defined the job. String. Mandatory. This argument is used by the Control-M/Server security mechanism and, under certain circumstances, cannot be modified. For more information, see the Security chapter and the description of the AuthorSecurity system parameter in GUI Server parameters.
FILE_PATH	Name of the library/directory in which the job script resides. String. Optional.
CMDLINE	Command string supplied when the job Task Type is Command. String. Optional.
HOSTID	Host name of an agent computer or name of a host group to which the job is submitted. String. Optional.
RUN_AS	Owner (user ID) associated with the job. This parameter is used by the Control-M/Server security mechanism. String. Mandatory.
MAXRERUN	Specifies the maximum number of reruns that can be performed for the job. Optional. Valid values: 0-99. Default: 0
TIMEFROM	Indicates the earliest time for submitting the job. Format is (hhmm). Optional.
TIMETO	Indicates the latest time for submitting the job. Format is (hhmm) or (>). Optional.
DUE_OUT	Time that the job is expected to finish. String. Optional.
PRIORITY	Indicates Control-M job priority. 2 alphanumeric characters (AA-99). Optional.

Parameter	Description
CRITICAL	Indicates that the job is a critical-path job in Control-M. Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
CYCLIC	Indicates whether the job is cyclic (to be run at regular intervals). Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
CYCLIC_TYPE	Determines the type of cyclic job: ▪ Interval ▪ IntervalSequence ▪ SpecificTimes Relevant for Control-M version 6.4.01 or later.
CYCLIC_TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Relevant for Control-M version 6.4.01 or later.
CYCLIC_INTERVAL_ SEQUENCE	A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas. Relevant for Control-M version 6.4.01 or later.
CYCLIC_TIMES_ SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.
INSTREAM_JCL (*embedded script)*	Defines a script exactly as it would be specified in a terminal for the specific computer and is part of the job definition. For Control-M for Distributed System for 6.4.01 or later an embedded script is marked with a &#10 indicator. For Control-M for z/OS version 6.2.00 or later, the XML file is formatted with multi\-ple lines which contain four digit numbers which state the length of the line, fol\-lowed by the line’s text.

Parameter	Description
USE_INSTREAM_JCL	Indicates whether the job should use the INSTREAM_JCL. **Valid values**: Y or N Relevant for jobs running in Control-M for z/OS version 6.2.00 or later and Control-M for Distributed System for 6.4.01 or later.
CONFIRM	Indicates whether the job must be manually confirmed by the Control-M/EM user before it runs. Optional. Valid values: ▪ 0 (Default) ▪ 1
AUTOARCH	Determines whether SYSDATA is to be archived. Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
INTERVAL	Specifies the length of time (in minutes) to wait between reruns of a job or between cyclic runs of a job. Integer. Optional. Default: 0.
OVERRIDE_PATH	Name of an alternate job script library/directory. String. Optional.
MAXWAIT	Number of extra days (beyond the original scheduling date) that the job is allowed to remain in the Active Jobs database awaiting execution. Integer. Optional.
DESCRIPTION	Free text description of the job. String. Optional.
DOCMEM	Name of the file containing job documentation. String. Optional.
DOCLIB	Name of a library or directory containing the job documentation file. String. Optional.
DAYS	Days of the month on which to order the job. String. Optional.
DAYS_AND_OR	Indicates the relationship between specified Days values and Weekdays values. Optional. Valid values: ▪ AND ▪ OR
WEEKDAYS	Days of the week on which to order the job. String. Optional.

Parameter	Description
DATE	Specific dates on which to order the job. String. **mmdd** format. Optional. For example, January 10 is written as: DATE="0110"
DAYSCAL	Name of a user-defined calendar used to specify a set of days. String. Optional.
WEEKSCAL	Name of a calendar to be used to validate specified weekdays on which to order the job. String. Optional.
CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the job. String. Optional.
RETRO	Indicates whether the job is scheduled for possible execution after its original scheduling date (odate) has passed. Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
SHIFT	Describes how to shift the scheduling date of the job. Optional. Valid values: ▪ IGNOREJOB ▪ PREVDAY ▪ NEXTDAY ▪ NOCONFCAL
SHIFTNUM	Number of days to shift the scheduling date of the job. Optional. Valid values: -62 to 62.
MAXDAYS	Maximum number of days to retain the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of AUTOARCH. Optional. Valid values: 00 – 98, or 99 to indicate that SYSDATA is retained for an unlimited number of days.
MAXRUNS	Maximum number of job runs to retain the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of AUTOARCH. Optional. Valid values: 000 – 998, or 999 to retain SYSDATA data for all runs.
RERUNMEM	Name of the JCL member to use when the job is automatically rerun. String. 1 - 8 characters. Optional.
RETEN_DAYS	(*z/OS only*) Number of days to retain the job in the History Jobs file. String. Optional.
RETEN_GEN	(*z/OS only*) Maximum number of generations of a job to keep in the History Jobs file String. Optional.

Parameter	Description
PREV_DAY	Flag to indicate whether job scheduling is shifted to a previous working day in the CONFCAL calendar. Optional. Valid values: ▪ Y ▪ N
IND_CYCLIC	Indicates whether the interval between further runs of a cyclic job is counted from the start or the end of the previous job run. Optional. Valid values: ▪ START ▪ END ▪ TARGET
RULE_BASED_CALENDAR_RELATIONSHIP	Relationship (AND|OR) between the specified Rule-Based Calendars and the job's own basic scheduling criteria. Optional. Valid values: ▪ AND ▪ OR
TAG_RELATIONSHIP	Relationship (AND|OR) between the specified Schedule Tag criteria and the job's own basic scheduling criteria. This parameter is relevant only for jobs in a SMART Folder. Optional. This parameter is for backward compatibility. Valid values: ▪ AND ▪ OR
SYSDB	Determines whether one or multiple data sets are used to catalogue sysdata. Optional. Valid values: ▪ 0 (Multiple-Default) ▪ 1 (Single)
PDSNAME	Name of a partitioned dataset (PDS) to be checked for free space. String. Optional.
MINIMUM	Minimum number of free partitioned dataset tracks required by the library specified for the PDSNAME parameter. Integer. Optional.
CATEGORY	Name of a Control-D report decollating mission category that must be scheduled under Control-D when the job is scheduled under Control-M. String. Optional.

Parameter	Description
PREVENTNCT2	(*z/OS only*) Prevents dataset cleanup before the original job run Optional. Valid values: ▪ Blank – Does not perform data set cleanup before the original job run. Default. ▪ N – Does not prevent cleanup. ▪ Y - Prevents data set cleanup. This value is not valid for started tasks. ▪ L (List) – Do not perform data set cleanup before the original job run. Do generate messages that would be required for CDG adjustment during restart. ▪ F (Flush) – Halt processing of the job if any data set cleanup error is detected (even if z/OS would not have stopped processing the job).
JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC	Months when the job can run. Valid values: ▪ 0 (Not run. Default) ▪ 1 (Run)
OPTION	Job output (OUTPUT) handling options. Optional. Valid values: ▪ Release ▪ Delete ▪ Copy ▪ Move ▪ File ▪ NewDest ▪ ChangeClass
PAR	Certain OPTION values (such as Release, NewDest) require additional information. String. Optional.
FROM	Limits the OUTPUT handling operation to only OUTPUTs from the specified class. String. Optional.
ADJUST_COND	Indicates whether to ignore prerequisite conditions normally set by predecessor jobs if the relevant predecessor jobs are not scheduled. This parameter is relevant only for jobs in a SMART Folder. Optional. Valid values: ▪ 0 (Do not ignore. Default.) ▪ 1 (Ignore relevant prerequisite conditions)

Parameter	**Description**
CREATION_USER	Name of the user who created the job. String. Optional.
CREATION_DATE	Date on which the job was created. String. Optional.
CREATION_TIME	Time at which the job was created. String. Optional.
CHANGE_USERID	Name of the user who last modified the job. String. Optional.
CHANGE_DATE	Date on which the job was last modified. String. Optional.
CHANGE_TIME	Time at which the job was last modified. String. Optional.
JOB_RELEASE	For internal use. Do not include this parameter in your defjob input file.
JOB_VERSION	For internal use. Do not include this parameter in your deffolder input file.
APPL_TYPE	Indicates the type of external application (for example, SAP or Oracle Applications) on which the external application job runs. String. Up to 10 characters. Mandatory for external application jobs.
APPL_VER	Version of the external application (for example, SAP or Oracle) on which the external application job runs. String. Up to 10 characters. Mandatory for external application jobs.
APPL_FORM	Predefined set of external application parameters that are displayed in the Job Properties team. String. Up to 30 characters. Mandatory for external application jobs.
CM_VER	Indicates the version of external Application Add-on (for example, SAP or Oracle Applications) that is installed in the Control-M installation. String. Up to 10 characters. Mandatory for external application jobs.
MULTY_AGENT	When selected, broadcasts job submission details to all agents in a specified Host Group. Optional. Valid values: ▪ Y – run as multi-agent job ▪ N – not run as multi-agent job. Default.
ACTIVE_FROM	(*z/OS only*) Indicates the start of a period of time during which the job or SMART Folder can be ordered. Optional. Date Format: YYYYMMDD
ACTIVE_TILL	(*z/OS only*) Indicates the end of the time interval during which the job or SMART Folder can be ordered. Optional. Date Format: YYYYMMDD

Parameter	Description	
TIMEZONE	Indicates the global time zone used to calculate the interval for time-related conditions. String. Optional.	
SCHEDULING_ ENVIRONMENT	(*z/OS only*) Indicates the JES2 workload management scheduling environment associated with the job. String. Optional.	
SYSTEM_AFFINITY	Identity of the system in which the job must be initiated and executed (in JES2). Identity of the processor on which the job must execute (in JES3). String. Optional.	
REQUEST_NJE_HOST	Specifies the host in the JES network on which the job is to execute. String. Optional.	
JOBISN	For internal use. String. Optional.	

RULE_BASED_CALENDAR_NAMES	Wrapper for the Rule-Based Calendars listed with the RULE_BASED_ CALENDAR_NAME parameter. Optional.
	`RULE_BASED_CALENDAR_NAMES RULE_BASED_CALENDAR_NAME="rbc1"`
	`        RULE_BASED_CALENDAR_NAME="rbc2"`	

	RULE_BASED_CALENDAR_NAME	Name of the Rule-Based Calendars that apply to the SMART Folder. Mandatory.
TAG_NAMES	Wrapper for the tags listed with the TAG_NAME parameter. Optional. `TAG_NAMES TAG_NAME="tag1" TAG_NAME="tag2"`	
	TAG_NAME	Name of the schedule tags that apply to the SMART Folder. Mandatory. This parameter is for backward compatibility.
INCOND	In condition. Optional. `INCOND NAME="Cond1" ODATE="ODAT" AND_OR="AND" OP="("`	
	NAME	Name of the In condition. String. Mandatory. 1 - 255 characters, case-sensitive.
	ODATE	Order date of the In condition. String. Mandatory. Default: ODAT
	AND_OR	Relationship between conditions. Optional. Valid values: ▪ AND (default) ▪ OR

Parameter	Description	
	OP	Parentheses indicating parts of the condition that are interpreted first. String. Optional.
OUTCOND	Out condition. Optional. `OUTCOND NAME="Job1" ODATE="ODAT" SIGN="ADD"`	
	NAME	Name of the Out condition. String. Mandatory. 1 - 255 characters, case-sensitive.
	ODATE	Order date of the Out condition. String. Mandatory. Valid values: Default: ODAT
	SIGN	Indicates whether to add or delete the condition. Mandatory. Valid values: ■ ADD (default) ■ DEL
VARIABLE	Wrapper for the Variable expression. Optional. `VARIABLE EXP="%%PARM1=%%TIME"`	
	EXP	The Variable expression. String. Mandatory. Example: %%PARM1=%%TIME.
QUANTITATIVE	Wrapper for the Quantitative resource. Optional. `QUANTITATIVE NAME="TAPEDRIVE" QUANT="1"`	
	NAME	Name of the quantitative resource. String. Mandatory. 1 - 20 characters, case-sensitive.
	QUANT	Quantity of the resource. String. Mandatory. Valid values: 0 – 9999. Default: 1

Parameter	Description	
Control	Wrapper for the Control resource. Optional. `Control NAME="Resc1" TYPE="E"`	
	NAME	Name of the Control resource. String. Mandatory. Valid values: 1-20 characters, case-sensitive, trailing blanks only.
	TYPE	Type of resource. Valid values: ▪ E (exclusive-default) ▪ S (shared)

SHOUT	Wrapper for the Shout message. Optional.
	`SHOUT WHEN="EXECTIME" DEST="workstation1" URGENCY="R"`
	`        MESSAGE="Jobcompleted OK." TIME="1015"`	

	WHEN	Condition under which the Shout message is sent. Mandatory. Valid values: ▪ OK (default) ▪ NOTOK ▪ RERUN ▪ LATESUB ▪ LATETIME ▪ EXECTIME
	DEST	Recipient of the shout message. String. Mandatory. Valid values: 1-16 characters, case-sensitive. Mandatory.
	URGENCY	Indicates the urgency of the Shout message. Mandatory. Valid values: ▪ R (regular-default) ▪ U (urgent) ▪ V (very urgent)
	MESSAGE	Text of the message. String. Mandatory. Valid values: 1 - 255 characters, spaces allowed.
	TIME	Time that the message is sent. String. Mandatory.

Parameter	Description	
STEP_RANGE	Step range in the job that can be used in an ON PGMST statement. Optional. `STEP_RANGE NAME="cleanup" FPGMS="Defrag" TPGMS=""`	
	NAME	Name for the range. 1-7 character string. Mandatory. Valid values: 1 - 7 characters. Only trailing blanks are allowed.
	FPGMS	Name of the program to be run. as the first program step in the range. 1 - 8 character string. Mandatory.
	FPROCS	Name of the procedure to be run. as the first procedure step in the range. 1-8 character string. Mandatory.
	TPGMS	Last program step in the range. 1-8 character string. Mandatory. Subparameter TO is optional. If blank, its value defaults to the last step in the job.
	TPROCS	Last procedure step in the range. 1-8 character string. Mandatory. Subparameter TO is optional. If blank, its value defaults to the last step in the job.
ON	Optional. `ON STMT="CODE" CODE="rt5" AND_OR="AND"><ON`	
	STMT	A character string containing a statement from the job script file. String. 1-132 characters. Mandatory for the On Statement/Code parameter.
	CODE	Return codes or statuses that can satisfy the step or code event criteria if returned upon termination of the specified job steps. String. Optional. Valid values: 1-255 characters.
	PGMS	Step in the program. String. Optional. Valid value: 1-8 characters.
	PROCS	Step in the process. String. Optional. Valid values:1-8 characters.
	AND_OR	Relationship between On statements. Optional. Valid values: ▪ AND ▪ OR

Parameter	Description	
DO	Specifies a status for the job based on conditions specified in an On statement. Optional. `DO ACTION="OK"`	
	ACTION	Mandatory. Valid values: ▪ OK (Changes the status of the job to OK) ▪ NOTOK (Changes the status of the job to NOTOK) ▪ RERUN (Reruns the job) ▪ SPCYC (Prevents further runs of a cyclic job)
DOSHOUT	Shout message wrapper. Optional. `DOSHOUT DEST="Wkstn2" URGENCY="R" MESSAGE="Job5 completed OK"`	
	DEST	Recipient of the Shout message. String. Mandatory. Valid values: 1 - 16 characters, case-sensitive.
	URGENCY	Urgency of the Shout message. Valid values: ▪ R (regular-default) ▪ U (urgent) ▪ V (very urgent)
	MESSAGE	Text of Shout message. String, 1 - 255 characters, spaces allowed. Mandatory.
DOCOND	Specifies prerequisite conditions to be added or deleted. Optional. `DOCOND NAME="Cond1" ODATE="ODAT" SIGN="ADD"`	
	NAME	Condition name. String, 1 - 20 characters, case-sensitive. Mandatory.
	ODATE	Condition date. String. Mandatory. Default: ODAT
	SIGN	Specifies whether to add or delete the condition. Valid values: ▪ ADD (default) ▪ DEL

Parameter	Description	
DOVARIABLE	Wrapper for the Variable expression. Optional. `DOVARIABLE EXP="%%PARM1=%%TIME"`	
	EXP	The Variable expression. String. Required. `%%PARM1=%%TIME`
DOFORCEJOB	Forces a specified job when the current job is performed. Optional. `DOFORCEJOB DSN="45446" FOLDER_NAME="Folder2" NAME="Job4" ODATE="ODAT"`	
	DSN	(*z/OS only*) Library for SMART Folder. String. Mandatory.
	FOLDER_NAME	Name of the SMART Folder to which the job belongs. String, 1-10 characters. Mandatory.
	NAME	Name of the job. String. Mandatory.
	ODATE	Original scheduling date for the job. String. Default: ODAT
DOOUTPUT	Handle job output (OUTPUT) when the job is done. Optional. `DOOUTPUT OPTION="Release" PAR="F" FROM=""`	
	OPTION	Output handling options. Mandatory. Valid values: [All platforms] ■ Release ■ Delete [not used with z/OS] ■ Copy ■ Move [*z/OS only*] ■ File ■ NewDest ■ ChangeClass
	PAR	Certain OPTION values require that you supply additional information (such as Release, NewDest). String. Optional.
	FROM	Limits the job output (OUTPUT) handling operation to only OUTPUTs from the specified class. String. Optional.

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="6"><b>DOIFRERUN</b></td><td colspan="2">Job steps to be executed during restart of a job. Available only at sites using Control-M/Restart. Optional.<pre>DOIFRERUN CONFIRM="0" FPGMS="step1" FPROCS="proc1"
        TPGMS="step5" TPROCS="proc3"</pre></td></tr>
<tr><td><i>CONFIRM</i></td><td>Indicates if job must be confirmed by user to be rerun.
Valid values:
▪ 0 (No confirmation. Default)
▪ 1 (Confirmation)</td></tr>
<tr><td><i>FPGMS</i></td><td>Step at which the job must be restarted. String. 1 - 8 characters. Mandatory.</td></tr>
<tr><td><i>FPROCS</i></td><td>Program step within the called procedure. String. 1 - 8 characters. Mandatory.</td></tr>
<tr><td><i>TPGMS</i></td><td>Step at which restarted job must terminate. String. 1 - 8 characters. Mandatory.</td></tr>
<tr><td><i>TPROCS</i></td><td>Program step within the called procedure. String. 1 - 8 characters.
Mandatory.</td></tr>
<tr><td rowspan="8"><b>DOMAIL</b></td><td colspan="2">Sends mail when the job run is complete. Optional.<pre>DOMAIL URGENCY="R" DEST="emuser@emuser.com"
        CC_DEST="barry@emuser.com" SUBJECT="OK"
        MESSAGE="Task completed OK."</pre></td></tr>
<tr><td><i>ATTACH_OUTPUT</i></td><td>Specifies at the job level whether the OUTPUT should be sent as an email attachment.</td></tr>
<tr><td><i>URGENCY</i></td><td>Urgency of the message.
Valid values:
▪ R (regular - Default)
▪ U (Urgent)</td></tr>
<tr><td><i>DEST</i></td><td>Recipient of the message. String. Mandatory.</td></tr>
<tr><td><i>CC_DEST</i></td><td>Additional recipient of the message. String. Optional.</td></tr>
<tr><td><i>SUBJECT</i></td><td>Brief text description of the message contents. String. Optional.</td></tr>
<tr><td><i>MESSAGE</i></td><td>Text of the message. String. Mandatory.</td></tr>
</table>

Parameter	Description	
DOCTBRULE	(*z/OS only*) Invokes a Control-M/Analyzer rule to be executed during the processing of a specific program step. Optional. `DOCTBRULE NAME="GOVTBAL" PAR="DOREPORT,10,%%ODATE"`	
	NAME	Name of the Control-M/Analyzer rule. String. Mandatory.
	PAR	Arguments that are passed to the Control-M/Analyzer rule. String. Optional. Maximum: 45 characters.
DOREMEDY	Opens a ticket in the Remedy Help Desk regarding the critical service.	
	URGENCY	The urgency level of the ticket that is opened in Remedy. Mandatory. Valid values are: ▪ L = Low (Default) ▪ M = Medium ▪ H = High ▪ U= Urgent ▪ C = Clear
	SUMMARY	A brief summary is displayed in Remedy. By default, a summary of the problem appears using variables. For more information on the field's characteristics, see .
	DESCRIPTION	A detailed description is displayed in Remedy. By default, a description of the problem appears using variables. For more information on the field's characteristics, see .

defjob input file example

The following example input file is used with the defjob utility:

```
<DEFJOB>

<JOB>

FOLDER_NAME="Tbl1"

FOLDER_DSN="2232"

JOBNAME="Job1"

FILE_NAME="Job1"

FILE_NAME="Job1"

SUBAPPLICATION="ACCT"

APPLICATION="App3"

DATACENTER="CTMNYC"

TASKTYPE="Command"

FOLDER_ORDER_ METHOD=""

FILE_PATH="JobLib1"

RUN_AS="Brad"

CREATED_BY="CTMEMUSER"

TIMEFROM="1210"

TIMETO="1310"

MAXRERUN="1"

INTERVAL="1"

PRIORITY="1"

CRITICAL="1"

CYCLIC="1"

CONFIRM="1"

DAYS="1,2,3"

DAYSCAL="Thurs">

<INCOND NAME="Cond1"/>

<OUTCOND NAME="Cond5"/>

<VARIABLE EXP="%%def=123"/>

<QUANTITATIVE NAME=""/>

<SHOUT WHEN="OK" DEST="COMP554" MESSAGE="Job
done." TIME="14:30"/>

<STEP_RANGE NAME="" FPGMS="1" FPROCS="1"
TPGMS="1" TPROCS="1"/>
```

```
<ON PGMS="" PROCS="" CODE="">

<DO ACTION="OK"/>

<DO ACTION="OK"/>

<DOVARIABLE EXP="%%def=123"/>

<DOSHOUT DEST="" MESSAGE=""/>

<DOFORCEJOB DSN="" FOLDER_NAME="Tbl1"
NAME="Job4"/>

<DOCTBRULE NAME="DOCT"/>

<DOOUTPUT OPTION="Release"/>

<DOIFRERUN FPGMS="1" FPROCS="1" TPGMS="1"
TPROCS="1"/>

<DOCOND NAME="COND4"/>

<DOMAIL DEST="COMP667" MESSAGE="Job done."/>

</ON>

</JOB>

</DEFJOB>
```

copydefjob

The copydefjob parameter of the emdef utility creates a new job definition in the Control-M/EM database that is similar to a specified existing definition. The original job and the copy must be in different data centers or SMART Folders. To run the copydefjob utility, see Running the copydefjob utility (on page 61).

Multiple jobs can be selected and copied using the * wildcard character. For an explanation of how wildcards function in the XML-based utilities, see Abbreviations and conventions (on page 10).

When copydefjob is invoked, it processes a file of arguments that specifies criteria for selecting one or more existing job processing definitions. The selected jobs are copied to the existing SMART Folder and/or data center specified in the arguments file. For more information, see copydefjob arguments file (on page 64).

Running the copydefjob utility

This procedure describes how to run the copydefjob utility, which enables you to create a new job definition in the Control-M/EM database that is similar to a specified existing definition.

➢ To run the copydefjob utility:

1. Do one of the following:
   a. Log on to a Control-M/EM account (*UNIX*).
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter either of the following commands:

- `emdef copydefjob [-u <user> [-p <password>] | -pf <password file>] -s <GUI Server Name> -arg <XML file name> [/a]`
- `emdef copydefjob [-USERNAME <user> [-PASSWORD <password>] | -PASSWORD_FILE <passwordFile>] -HOST <guiServerName> -ARG_FILE <XML file name> [/a]`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the copydefjob parameters and switches, see copydefjob parameters (on page 62) and copydefjob switches (on page 63).

copydefjob parameters

The following table lists the copydefjob paramters:

Parameter	Description
<user name>	Control-M/EM user name.
<password>	Control-M/EM user password.

<password file name>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=*<userName>*
	password=*<password>*

<GUI Server Name>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<XML file name>	The path and name of the XML file containing the defjob specifications. For more information, see Control-M Variable facility .

copydefjob switches

The following table lists optional switches for the copydefjob utility:

Switch	Description
?	Displays utility's description and available options.
/a	The /a switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The /a switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to **2** or **3.**
/v	Used to receive verbose messages.

copydefjob arguments file

As the utility runs, the copydefjob arguments file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

Arguments are used as selection criteria to determine which jobs to copy. Arguments are written to the copydefjob argument file. The arguments files that you create with the copydefjob utility are written in XML format and saved in a text file. The format in which this file must be written is described on the following pages.

When this file is invoked, job processing definitions are exported from the Control-M/EM database.

Each parameter that you specify must have a FROM value. This value is used as a search criteria for selecting jobs to copy.

The copydefjob utility can use only simple job parameters as search and replace criteria. Complex parameters, such as the name of an In Condition or the degree of urgency of a Do Shout parameter, cannot be used as search criteria or modified with the copydefjob utility.

The following rules apply to the copydefjob utility arguments file:

- More than one job can be specified in the arguments file.
- The arguments file is case-sensitive.
- All parameter values (such as strings or digits) must be enclosed in quotation marks (for example, JOBNAME="Job1").
- Only one COPYJOB parameter can be used in an arguments file.
- The COPYJOB parameter must contain only one of each job parameter. Many job parameters are optional.
- Multiple values can be specified for TO and FROM by using the * wildcard character. For an explanation of how wildcards function in the XML-based utilities, see Abbreviations and conventions (on page 10).
- If any FROM value contains a * and the corresponding TO value contains a *, the * in the TO value represents the same information the * in the FROM value.
- Changing the data center name or the SMART Folder name imports the copy of the job into a data center or SMART Folder different from the one in which the original job was located.

Most job definition parameters are optional. However,

- if you specify any parameters, the FROM subparameter is mandatory and the TO subparameter is optional.
- if a FROM value is specified without a TO value, it is used as a filter criterion.
- if a TO value is specified, it indicates the new value of the parameter.

For more information on the arguments file parameters for the copydefjob utility, see copydefjob arguments file parameters (on page 65), and copydefjob arguments file examples (on page 80).

copydefjob arguments file parameters

The following table lists input file parameters for the copydefjob utility:

Parameter	Description	
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the **.dtd** file.		
COPYJOB	These tags indicate the start and end of the COPYJOB argument. Only criteria that are located between the tags are considered to be part of the argument.	
FOLDER_NAME	Name of the SMART Folder to which the job belongs. At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN `FOLDER_NAME FROM="Tbl5NYC" TO="Tbl7NYC"`	
	FROM	Name of the SMART Folder specified in the job processing definition that is being copied. String. Mandatory.
	TO	The folder name in the job processing definition copy. String. Optional.
FOLDER_DSN	(*z/OS® only*) Name of the library that contains the SMART Folder. Mandatory. At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN `FOLDER_DSN FROM="Lib1" TO="Lib1_COPY"`	
	FROM	Name of the library containing the SMART Folder in the job processing definition that is being copied. String. Mandatory.
	TO	Name of the library in the job processing definition copy. String. Optional.

Parameter	Description	
DATACENTER	Name of the Control-M installation to which the job belongs. At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN `DATACENTER FROM="EM_Montreal" TO="EM_Paris"`	
	FROM	Name of the Control-M installation to which the job being copied belongs. Mandatory.
	TO	Name of the Control-M installation to which the job copy belongs. Optional.

Parameter	Description	
JOBNAME	Name of the job processing definition. JOBNAME FROM="Job3"	
	FROM	String. Mandatory.
FILE_NAME	Name of the file that contains the job script. MEMNAME FROM="Mem3"	
	FROM	String. Mandatory.
SUB_APPLICATION	Name of the group to which the job belongs. Used as a descriptive name for related groups of jobs. SUB_APPLICATION FROM="Grp_HR"	
	FROM	String. Mandatory.
APPLICATION	Name of the application to which the job's group belongs. Used as a descriptive name for related groups of jobs. APPLICATION FROM="App3""	
	FROM	String. Mandatory.
TASKTYPE	Type of the job (task) to be performed by Control-M. TASKTYPE FROM="Detached"	
	FROM	Mandatory. Valid values: ▪ Job ▪ Detached ▪ Command ▪ Dummy (*z/OS only*) Valid values: ▪ Started_Task ▪ Cyclic_Job ▪ Cyclic_Task ▪ Emergency_Job ▪ Emergency_Cyclic_Job ▪ Emergency_Task ▪ Emergency_Cyclic_Task

Parameter	Description	
FOLDER _ORDER_ METHOD	Optional. `FOLDER_ORDER_ METHOD FROM="Job3"`	
	FROM	String. Mandatory.
CREATED_BY	Control-M/EM user who defined the job. String. Optional. `CREATED_BY FROM="emuser"` This argument is used by the Control-M/Server security mechanism and, under certain circumstances, cannot be modified. For more information, see the Security chapter and the description of the AuthorSecurity system parameter in GUI Server parameters.	
	FROM	String. Mandatory.
FILE_PATH	Name of the library/directory in which the job script resides. String. Optional. `FILE_PATH FROM="Mem1"`	
	FROM	String. Mandatory.
CMDLINE	Command string supplied when the job Task Type is **Command**. Optional. `CMDLINE FROM="C:\Format"`	
	FROM	String. Mandatory.
HOSTID	Host name of an agent computer or name of a host group to which the job is submitted. Optional. `HOSTID FROM="Com3"`	
	FROM	String. Mandatory.
RUN_AS	Owner (user ID) associated with the job. This parameter is used by the Control-M/Server security mechanism. Optional. `OWNER FROM="emuser"`	
	FROM	String. Mandatory.
MAXRERUN	Specifies the maximum number of reruns that can be performed for the job. `MAXRERUN FROM="1"` Valid values: **0-99**. Default: **0**	
	FROM	String. Mandatory.

Parameter	Description	
TIMEFROM	Indicates the earliest time for submitting the job. `TIMEFROM FROM="1430"`	
	FROM	String. Mandatory.
TIMETO	Indicates the latest time for submitting the job. `TIMETO FROM="1600"`	
	FROM	String. Mandatory.

DUE_OUT	`Time that the job is expected to finish.`
	`DUE_OUT FROM="1500"`	

	FROM	String. Mandatory.

Parameter	Description	
PRIORITY	Indicates Control-M job priority. `PRIORITY FROM="AA"`	
	FROM	String. Mandatory.
CRITICAL	Indicates that the job is a critical-path job in Control-M. `CRITICAL FROM="0"`	
	FROM	Mandatory. Valid values: ▪ **0** (Default) ▪ **1**
CYCLIC	Indicates if the job is cyclic (to be rerun at regular intervals). Optional. `CYCLIC FROM="0"`	
	FROM	Mandatory. Valid values: ▪ **0** (Default) ▪ **1**
CYCLIC_TYPE	Determines the type of cyclic job: ▪ Interval ▪ IntervalSequence ▪ SpecificTimes Relevant for Control-M version 6.4.01 or later.	
CYCLIC_TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Relevant for Control-M version 6.4.01 or later.	
CYCLIC_INTERVAL_SEQUENCE	A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas. Relevant for Control-M version 6.4.01 or later.	
CYCLIC_TIMES_SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.	

Parameter	Description	
CONFIRM	Indicates that the job must be manually confirmed by the Control-M/EM user before it runs. `CONFIRM FROM="0"`	
	FROM	Mandatory. Valid values: ▪ **0** (Default) ▪ **1**
AUTOARCH	Determines whether SYSDATA is to be archived. `AUTOARCH FROM=0"`	
	FROM	Mandatory. Valid values: ▪ **0** (Default) ▪ **1**
INTERVAL	Specifies the length of time (in minutes) to wait between reruns of a job or between cyclic runs of a job. Integer. Optional. `INTERVAL FROM="3"`	
	FROM	String. Mandatory.
OVERRIDE_PATH	Name of an alternate job script library/directory. String. `OVERRIDE PATH FROM="lib3"`	
	FROM	String. Mandatory.
MAXWAIT	Number of extra days (beyond the original scheduling date) that the job is allowed to remain in the Active Jobs database awaiting execution. Integer. `MAXWAIT FROM="4"`	
	FROM	String. Mandatory.
DESCRIPTION	Free text description of the job. String. `DESCRIPTION FROM="data backup from 120399"`	
	FROM	String. Mandatory.
DOCMEM	Name of the file containing job documentation. String. `DOCMEM FROM="mem4"`	
	FROM	String. Mandatory.

Parameter	Description	
DOCLIB	Name of a library or directory containing the job documentation file. String. Mandatory. `DOCLIB FROM="AcctFiles"`	
	FROM	String. Mandatory.
DAYS	Days of the month on which to order the job. String. Optional. `DAYS FROM="ALL"`	
	FROM	String. Mandatory.
DAYS_AND_OR	Indicates the relationship between specified Days values and Weekdays values. Optional. `DAYS_AND_OR FROM="AND"`	
	FROM	String. Mandatory.
WEEKDAYS	Days of the week on which to order the job. String. Optional. `WEEKDAYS FROM="1,2,4"`	
	FROM	String. Mandatory.
DATE	Specific dates on which to order the job. String. MMDD format. Optional. `DATE FROM="0312"`	
	FROM	String. Mandatory. Dates can be written in mmdd format. There is no delimiter between dates. For example, January 10 is written in this manner: DATE="0110"
DAYSCAL	Name of a user-defined calendar used to specify a set of days. String. Optional. `DAYSCAL FROM="shipping"`	
	FROM	String. Mandatory.
WEEKSCAL	Name of a calendar to be used to validate specified weekdays on which to order the job. String. Optional. `WEEKSCAL FROM="2"`	
	FROM	String. Mandatory.
CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the job. String. `CONFCAL FROM="cal99" TO="cal00"`	

Parameter	**Description**	
	FROM	String. Mandatory.
RETRO	Indicates whether the job is scheduled for possible execution after its original scheduling date (odate) has passed. `RETRO FROM="0"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
SHIFT	Describes how to shift the scheduling date of the job. `SHIFT FROM="PREVDAY"`	
	FROM	Mandatory. Valid values: ▪ IGNOREJOB ▪ PREVDAY ▪ NEXTDAY ▪ NOCONFCAL
SHIFTNUM	Number of days to shift the scheduling date of the job. `SHIFTNUM FROM="-10"`	
	FROM	String. Mandatory.
MAXDAYS	Maximum number of days to retains the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of AUTOARCH. String. Optional. `MAXDAYS FROM="07"`	
	FROM	Integer. Mandatory.
MAXRUNS	Maximum number of job runs to retains the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of AUTOARCH. String. Optional. `MAXRUNS FROM="100"`	
	FROM	String. Mandatory.
RERUNMEM	Name of the JCL member to use when the job is automatically rerun. String. 1-8 characters. Optional. `RERUNMEM FROM="Mem45"`	
	FROM	String. Mandatory.

Parameter	Description	
RETEN_DAYS	(*z/OS only*) Number of days to retain the job in the History Jobs file. String. Optional. `RETEN_DAYS FROM="5"`	
	FROM	String. Mandatory.
RETEN_GEN	(*z/OS only*) Maximum number of generations of the job to keep in the History Jobs file. String. `RETEN_GEN FROM="3"`	
	FROM	String. Mandatory.
TASK_CLASS	Job class for the task. `TASK_CLASS FROM="Distribution"`	
	FROM	Mandatory. Valid values: ▪ Distribution ▪ Decollation
PREV_DAY	Flag to indicate whether job scheduling is shifted to a previous working day in the CONFCAL calendar. Optional. `PREV_DAY FROM="N"`	
	FROM	Mandatory. Valid values: ▪ Y ▪ N
IND_CYCLIC	Indicates whether the interval between further runs of a cyclic job is counted from the start or the end of the previous job run. Optional. `IND_CYCLIC FROM="START"`	
	FROM	Mandatory. Valid values: ▪ START ▪ END
RULE_BASED_CALENDAR_RELATIONSHIP	Relationship (AND|OR) between the specified Rule-Based Calendar and the job's own basic scheduling criteria. Optional. `RULE_BASED_CALENDAR_RELATIONSHIP FROM="AND"`	

Parameter	Description	
	FROM	Mandatory. Valid values: ▪ AND ▪ OR
TAG_RELATIONSHIP	Relationship (AND|OR) between the specified Schedule Tag criteria and the job's own basic scheduling criteria. This parameter is relevant only for jobs in a SMART Folder. Optional. This parameter is for backward compatibility. `TAG_RELATIONSHIP FROM="AND"`	
	FROM	Mandatory. Valid values: ▪ AND ▪ OR
SYSDB	Determines whether one or multiple data sets are used to catalogue sysdata. `SYSDB FROM="1" TO="0"`	
	FROM	Mandatory. Valid values: ▪ 0 (Multiple -Default) ▪ 1 (Single)
PDSNAME	Name of a partitioned dataset (PDS) to be checked for free space. String. Optional. `PDSNAME FROM="Lib_3"`	
	FROM	String. Mandatory.
MINIMUM	Minimum number of free partitioned dataset tracks required by the library specified for the PDSNAME parameter. Integer. Optional. `MINIMUM FROM="5"`	
	FROM	Integer. Mandatory.
CATEGORY	Name of a Control-D report decollating mission category that must be scheduled under Control-D when the job is scheduled under Control-M. String. Optional. `CATEGORY FROM="DAILY"`	
	FROM	String. Mandatory.

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="2"><em>PREVENTNCT2</em></td><td colspan="2">(<em>z/OS only</em>) Prevents dataset cleanup before the original job run. Optional.
Valid values:
▪ Blank – Does not perform data set cleanup before the original job run. Default.
▪ N – Does not prevent cleanup.
▪ Y - Prevents data set cleanup. This value is not valid for started tasks.
▪ L (List) – Do not perform data set cleanup before the original job run. Do generate messages that would be required for CDG adjustment during restart.
▪ F (Flush) – Halt processing of the job if any data set cleanup error is detected (even if z/OS would not have stopped processing the job).
<code>PREVENTNCT2 FROM="1"</code></td></tr>
<tr><td>FROM</td><td>Mandatory. Valid values:
▪ 0
▪ 1</td></tr>
<tr><td rowspan="2"><em>JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,
OCT, NOV, DEC</em></td><td colspan="2">Months when the job can run. Optional.
<code>JAN FROM="0"</code></td></tr>
<tr><td>FROM</td><td>Mandatory. Not including a month is the same as including a month having the value 0. Valid values:
▪ 0 (Default)
▪ 1</td></tr>
<tr><td rowspan="2"><em>OPTION</em></td><td colspan="2">Job output (OUTPUT) handling options.
<code>OPTION FROM="Copy"</code></td></tr>
<tr><td>FROM</td><td>Mandatory. Valid values:
▪ Release
▪ Delete
▪ Copy
▪ Move
▪ File
▪ NewDest
▪ ChangeClass</td></tr>
</table>

Parameter	Description	
PAR	Certain OPTION FROM values (such as Release, NewDest) require additional information. The PAR parameter holds this information as a string. `PAR FROM="mem3.log"`	
	FROM	String. Mandatory.
FROM	Limits the OUTPUT handling operation to OUTPUTs from the specified class. Optional. `FROM FROM="5"`	
	FROM	String. Mandatory.
ADJUST_COND	Indicates whether to ignore prerequisite conditions normally set by predecessor jobs if the relevant predecessor jobs are not scheduled. This parameter is relevant only for jobs in a SMART Folder. Optional. Valid values: ▪ 0 (Do not ignore. Default.) ▪ 1 (Ignore relevant prerequisite conditions)	
	FROM	String. Mandatory.
APPL_TYPE	Indicates the type of external application (for example, SAP or Oracle) on which the external application job runs. Mandatory for external application jobs. `APPL_TYPE FROM="SAP"`	
	FROM	Mandatory. String. Up to 10 characters.
APPL_VER	Version of the external application (for example, SAP or Oracle) on which the external application job runs. Mandatory for external application jobs. `APPL_VER FROM="4.6"`	
	FROM	Mandatory. String. Up to 10 characters.
APPL_FORM	Predefined set of external application parameters that are displayed in the Job Properties team. Mandatory for external application jobs. `APPL_FORM FROM="Default SAP 4.6"`	
	FROM	Mandatory. String. Up to 30 characters.
CM_VER	Indicates the version of external Application Add-on (for example, SAP or Oracle) that is installed in the Control-M installation. Mandatory for external application jobs. `CM_VER FROM="6.1.00"`	

Parameter	Description	
	FROM	Mandatory. String. Up to 10 characters.
MULTY_AGENT	When selected, broadcasts job submission details to all agents in a specified Host Group. Optional. `MULTY_AGENT FROM="N"`	
	FROM	Mandatory. Valid values: ▪ Y – run as multi-agent job ▪ N – not run as multi-agent job. Default.
ACTIVE_FROM	(*z/OS only*) Indicates the start of a period of time during which the job or SMART Folder can be ordered. Optional. `ACTIVE_FROM FROM="20070315"`	
	FROM	Mandatory. Date Format: YYYYMMDD
ACTIVE_TILL	(*z/OS only*) Indicates the end of a period of time during which the job or SMART Folder can be ordered. Optional. `ACTIVE_TILL FROM="20070315"`	
	FROM	Mandatory. Date Format: YYYYMMDD
TIMEZONE	Indicates global time zone used to calculate the interval for time-related conditions. Optional. `TIMEZONE FROM="EST"`	
	FROM	Mandatory. String. Default: GMT
SYSTEM_AFFINITY	Identity of the system in which the job must be initiated and executed (in JES2). Identity of the processor on which the job must execute (in JES3). Optional. String.	
	FROM	String. Mandatory. `SYSTEM_AFFINITY FROM="SYS3"`
REQUEST_NJE _HOST	Specifies the host in the JES network on which the job is to execute.	
	FROM	String. Mandatory. `REQUEST_NJE_HOST FROM="OS5"`

Parameter	Description	
SCHEDULING_ENVIRONMENT	(*z/OS only*) Indicates the JES2 workload management scheduling environment that is to be associated with the job.	
	FROM	String. Mandatory. `SCHEDULING_ENVIRONMENT FROM="SCHD2"`
CREATION_USER	Name of the user who created the job. String. `CREATION_USER FROM="emuser"`	
	FROM	String. Mandatory.
CREATION_DATE	Date on which the job was created. String. `CREATION_DATE FROM="1212"`	
	FROM	String. Mandatory.
CREATION_TIME	Time at which the job was created. String. `CREATION_TIME FROM="1230"`	
	FROM	String. Mandatory.
CHANGE_USERID	Name of the user that last modified the job. String. `CHANGE_USERID FROM="emuser"`	
	FROM	String. Mandatory.
CHANGE_DATE	Date that the job was last modified. String. `CHANGE_DATE FROM="1204"`	
	FROM	String. Mandatory.
CHANGE_TIME	Time that the job was last modified. String. `CHANGE_TIME FROM="1650"`	
	FROM	String. Mandatory.

copydefjob arguments file examples

The following example argument files are used with the copydefjob utility:

Copy selected jobs and change parameter values in the copies

This copydefjob arguments file copies job processing definitions in the Tbl5NYC folder if FOLDER_DSN is Lib1 and JOBNAME is Job3. In the copy, the FOLDER_DSN value is changed to Lib1_COPY.

```
<COPYJOB>
<FOLDER_NAME FROM="Tbl5NYC"/>
<FOLDER_DSN FROM="Lib1" TO="Lib1_COPY"/>
<JOBNAME FROM="Job3"/>
</COPYJOB>
```

Copy all jobs in one folder to another folder

Copies of all jobs in the Tbl5NYC folder to the Tbl7LA folder.

```
<COPYJOB>
<FOLDER_NAME FROM="Tbl5NYC" TO="Tbl7LA"/>
</COPYJOB>
```

Copy all cyclic jobs with a similar job name

Copies of all cyclic jobs in the GrpAcct group that have a jobname beginning with the string Acct from FOLDER_DSN 23Y to FOLDER_DSN 14G.

```
<COPYJOB>
<FOLDER_DSN FROM="23Y" TO="14G">
<SUBAPPLICATION FROM="GrpAcct"/>
<CYCLIC FROM="1"/>
<JOBNAME="Acct*" />
</COPYJOB>
```

Copy jobs in a folder to a SMART Folder

You can copy the jobs in a folder to a SMART Folder using the copydefjob utility.

- Using Control-M Workload Automation, define a SMART Folder containing no jobs.
- Write the SMART Folder to the Control-M/EM database.
- Create a copydefjob arguments file in which jobs in a folder are copied to the SMART Folder that you created:
- fpen a text editor. Format the file using the specifications in the copdefjob arguments file.
- Specify the Control-M installation in which the jobs to be copied reside using the DATACENTER parameter:
- DATACENTER FROM="CTM_Name"

- Specify that the Folder Name value of the jobs changes from the name of the folder to the name of the SMART Folder with the following tag:
- FOLDER_NAME FROM="Sched_Tbl_Name" TO="Grp_Sched_Tbl_Name"
- Save and close the file.
- At the command line, enter the copydefjob utility command that uses the file that you created in the previous step:
- `copydefjob -u <emUser> -p <emPass> -s <guiServerName> -arg <argFileName>`
- Use the Folder Manager in Control-M Workload Automation to upload the SMART Folder to the Control-M/Server.
- In Control-M Workload Automation, order the SMART Folder.
- In Control-M/EM Workload Automation, verify that the SMART Folder now contains the jobs that were copied to it.
- If you do not need the original folder, you can delete it.

The following arguments file copies all jobs in the RegFolder folder in the ctm600 data center to the GrpSFolder SMART Folder.

```
<COPYJOB>

<DATACENTER FROM="ctm700"/>

<FOLDER_NAME FROM="RegFolder" TO="GrpSFolder"/>

</COPYJOB>
```

deldefjob

The deldefjob utility deletes specified job processing definitions in the Control-M/EM database. To run the deldefjob utiltity, see Running the deldefjob utility (on page 81).

When deldefjob is invoked, it processes a specified file of arguments in XML format. This file contains statements that identify existing job processing definitions. The identified definitions are deleted from the Control-M/EM database. For more information, see deldefjob arguments file (on page 83).

Running the deldefjob utility

This procedure describes how to run the deldefjob utility, which enables you to delete specified job processing definitions in the Control-M/EM database.

➢ To run the deldefjob utility:

1. Do one of the following:
   a. Log on to a Control-M for Databases account (*UNIX*)
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter either of the following commands:

- `emdef deldefjob [-u <user> [-p <password>] | -pf <password file>] -s <GUI Server Name> -arg <XML file name> [/a]`
- `emdef deldefjob [-USERNAME <user> [-PASSWORD <password>] | -PASSWORD_FILE <password file>] -HOST <GUI Server Name> -ARG_FILE <XML file name> [/a]`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the deldefjob parameters and switches, see deldefjob parameters (on page 82) and deldefjob switches (on page 82).

deldefjob parameters

The following table lists the deldefjob utility parameters:

Parameter	Description
<user name>	Control-M/EM user name.
<password>	Control-M/EM user password.

<password file name>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=*<userName>*
	password=*<password>*

<GUI Server Name>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<XML file name>	The path and name of the XML file containing the defjob specifications. For more information, see Control-M Variable facility .

deldefjob switches

The following table lists optional switches for the deldefjob utility:

Switch	Description
?	Displays utility's description and available options.
/v	Used to receive verbose messages.

deldefjob arguments file

The following rules apply to the deldefjob arguments file:

- More than one job can be specified in a **deldefjob** file.
- The arguments file is case-sensitive.
- More than one PARAM parameter can be used in a TERM statement.
- The relationship between PARAM parameters in a TERM statement is AND.
- The default relationship between TERM statements is OR.
- All parameter values must be enclosed in quotation marks (for example, JOBNAME="Job1").

The deldefjob arguments file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

For more information on the arguments file parameters for the deldefjob utility, see deldefjob arguments file parameters (on page 83), and deldefjob arguments file example (on page 84).

deldefjob arguments file parameters

The following table lists the input file parameters for the deldefjob utility:

Parameter	Description	
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the **.dtd** file.		
TERMS	These tags indicate the start and end of the TERM tags. Only criteria that are located between these tags are considered to be part of the argument.	
TERM	The TERM tags indicate the start and end of a group of selection criteria for a job or jobs that are to be deleted. Only PARAM tags that are located between the TERM tags are considered to be part of the TERM argument.	
	REL	Optional. Relationship between terms. Valid values: ▪ AND ▪ OR (default)

Parameter	Description	
PARAM	The selection criteria parameter used to determine job definitions to be deleted. More than one PARAM can be specified. Mandatory. `PARAM NAME="DATACENTER" OP="EQ" VALUE="Center1"`	
	NAME	String. Mandatory. Name of any job processing parameter. At least one of the following SMART Folder parameters must be included in the arguments file: ▪ DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN
	OP	Mandatory. Valid values: ▪ EQ – Equal ▪ NEQ – Not equal ▪ NOTIN – Does not contain ▪ LIKE – Mask or pattern using wildcards
	VALUE	String. Mandatory. Valid value for the specified job processing parameter.

deldefjob arguments file example

The following example argument files are used with the deldefjob utility:

Delete definitions with the same job name

Delete job processing definitions with the name Job5 from the EM5NY data center.

```
<TERMS>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="EM5NY"/>

<PARAM NAME="JOBNAME" OP="EQ" VALUE="Job5"/>

</TERM>

</TERMS>
```

Delete definitions that satisfy one or both of two criteria

Delete job processing definitions that satisfy either of the following criteria:

- The data center name is **Data1** and the jobname begins with the letter **J.**

- or -

- The jobname is **Job5** and the job is not cyclic.

```
<TERMS>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>

<PARAM NAME="JOBNAME" OP="LIKE" VALUE="J*"/>

</TERM>

<TERM>

<PARAM NAME="JOBNAME" OP="EQ" VALUE="Job5"/>

<PARAM NAME="CYCLIC" OP="EQ" VALUE="0"/>

</TERM>

</TERMS>
```

Delete definitions that meet multiple criteria

Delete definitions for cyclic jobs in the EM5NY data center that are scheduled to run in January, February, and March.

```
<TERMS>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="EM5NY"/>

<PARAM NAME="CYCLIC" OP="EQ"
VALUE="1"/>

<PARAM NAME="JAN" OP="EQ" VALUE="1"/>

<PARAM NAME="FEB" OP="EQ" VALUE="1"/>

<PARAM NAME="MAR" OP="EQ" VALUE="1"/>

</TERM>

</TERMS>
```

duplicatedefjob

The duplicatedefjob utility makes a copy of an existing job definition in the same data center and SMART Folder. Elements of the copy can be changed. To run the duplicatedefjob utility, see Running the duplicatedefjob (on page 86).

Multiple jobs can be selected and copied using the * wildcard character. For an explanation of how wildcards function in XML-based utilities, see .Abbreviations and conventions (on page 10)

When duplicatedefjob is invoked, it processes a specified file of arguments in XML format. This file contains statements that identify existing job processing definitions. The identified definitions are copied, changes to the copy (if requested) are made, and the copy is stored in the Control-M/EM database. For more information, see duplicatedefjob arguments file (on page 88).

Running the duplicatedefjob

The following procedure describes how to run the duplicatedefjob utility, which enables you to copy an existing job definition in the same data center and SMART Folder.

➢ To run the duplicatedefjob utility:

- Enter either of the following commands:
  - `emdef duplicatedefjob [-u <user> [-p <password>] | -pf <password file>] -s <GUI Server Name> -arg <XML file name> [/a]`
  - `emdef duplicatedefjob [-USERNAME <user> [-PASSWORD <password>] | -PASSWORD_FILE <password file>] -HOST <GUI Server Name> -ARG_FILE <XML file name> [/a]`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the duplicatedefjob parameters and switches, see duplicatedefjob parameters (on page 87) and duplicatedefjob switches (on page 87).

duplicatedefjob parameters

The following table describes the parameters of the duplicatedefjob utility:

Parameter	Description
<user name>	Control-M/EM user name.
<password>	Control-M/EM user password.

<password file name>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName*>
	password=<*password*>

<GUI Server Name>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<XML file name>	The path and name of the XML file containing the defjob specifications. For more information, see Control-M Variable facility .

duplicatedefjob switches

The following table describes an optional switch for the duplicatedefjob utility:

Switch	Description
?	Displays utility's description and available options.
/a	The /a switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The /a switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to 2 or 3.
/v	Used to receive verbose messages.

duplicatedefjob arguments file

Each arguments file parameter that you specify must have a FROM subparameter. The FROM value is used as a search criteria for selecting jobs to copy. For example, JOBNAME FROM="Job2" copies all jobs with the JobName Job2.

The TO subparameter, which is optional, is used to change the value of the parameter. For example, **JOBNAME FROM="Job2" TO="Job2B**" modifies all jobs with JobName Job2 so that they now have JobName Job2B.

Currently, the duplicatedefjob utility can use only simple job parameters as search and replace criteria. Complex parameters, such as the name of an In Condition parameter or the degree of urgency of a Do Shout parameter, cannot be used as search criteria or modified with the duplicatedefjob utility.

The following rules apply to the arguments file for the duplicatedefjob utility:

- More than one job can be specified in a duplicatedefjob file.
- The arguments file is case-sensitive.
- All parameter values must be enclosed in quotation marks (for example, JOBNAME="Job1").
- Only one DUPLICATEJOB parameter can be used in the arguments file. This parameter must not contain more than one instance of each job parameter.
- Multiple values can be specified for TO and FROM by using the * wildcard character. For an explanation of how wildcards function in the XML-based utilities, see Abbreviations and conventions (on page 10).
- If any FROM value contains *, and the corresponding TO value contains *, the * in the TO value expresses the same information as the * in the FROM value.
- Changing the data center name or SMART Folder name causes the copy of the job scheduling definition to be imported into the specified data center or SMART Folder.

Most job definition parameters in the arguments file are optional. However

- For each parameter that is specified, the FROM subparameter is mandatory and the TO subparameter is optional.
- When FROM is specified without a TO, the FROM value is used as a filter criterion.
- When TO is included, it specifies the value to which the parameter is set.

The duplicatedefjob arguments file is checked and processed. If the file contains errors, a message is displayed specifying the lines with the errors.

For more information on the arguments file parameters for the duplicatedefjob utility, see duplicatedefjob arguments file parameters (on page 89), and duplicatedefjob arguments file example (on page 109).

duplicatedefjob arguments file parameters

The following table list the arguments file parameters:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td colspan="3">The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the .dtd file.</td></tr>
<tr><td>DUPLICATEJOB</td><td colspan="2">The DUPLICATEJOB tags indicate the start and end of a group of selection criteria for a job or jobs that are to be copied. Only criteria that are located between the DUPLICATEJOB tags are considered to be part of the duplicatedefjob parameters.</td></tr>
<tr><td rowspan="2">FOLDER_NAME</td><td colspan="2">Name of the SMART Folder to which the job belongs. Mandatory.
At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN
FOLDER_NAME FROM="Tbl5NYC"</td></tr>
<tr><td>FROM</td><td>String. Mandatory.</td></tr>
</table>

Parameter	Description	
FOLDER_DSN	(*z/OS only*) Name of the library that contains the SMART Folder. Mandatory. ▪ A TO subparameter cannot be specified for this parameter. ▪ At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN `FOLDER_DSN FROM="Lib1"`	
	FROM	String. Mandatory.
DATACENTER	Name of the Control-M installation to which the job belongs. Mandatory. ▪ A TO parameter cannot be specified for this parameter. ▪ At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME,FOLDER_DSN `DATACENTER FROM="CTMNYC"`	
	FROM	String. Mandatory.
FOLDER_ ORDER_ METHOD	String. Mandatory. A TO parameter can be specified and modified for this parameter. `FOLDER_ORDER_ METHOD FROM="Job3"`	
	FROM	String. Mandatory.
JOBNAME	Name of the job processing definition. Optional. `JOBNAME FROM="Job3" TO="Job3_COPY"`	
	FROM	Mandatory.
	TO	Optional.
FILE_NAME	Name of the file that contains the job script. Optional. `FILE_NAME FROM="Mem3" TO="Mem7"`	
	FROM	Mandatory.
	TO	Optional.

Parameter	Description	
SUB_APPLICATION	Name of the group to which the job belongs. Used as a descriptive name for related jobs. Optional. `GROUP FROM="Grp_HR" TO="Grp_ACCT"`	
	FROM	Mandatory.
	TO	Optional.
APPLICATION	Name of the application to which the job's group belongs. Used as a descriptive name for related jobs. Optional. `APPLICATION FROM="App3" TO="App1"`	
	FROM	String. Mandatory.
	TO	String Optional.

Parameter	Description	
TASKTYPE	Type of the job (task) to be performed by Control-M. Optional. `TASKTYPE FROM="Detached" TO="Job"`	
	FROM	Mandatory. Valid values: ▪ Job ▪ Detached ▪ Command ▪ Dummy (*z/OS only*) Valid values: ▪ Started_Task ▪ Cyclic_Job ▪ Cyclic_Task ▪ Emergency_Job ▪ Emergency_Cyclic_Job ▪ Emergency_Task ▪ Emergency_Cyclic_Task
	TO	Optional. Valid values: Same as mandatory FROM values. (*z/OS only*) Valid values: Same as z/OS FROM values.
CREATED_BY	Control-M/EM user who defined the job. String. Optional. `CREATED_BY FROM="emuser" TO="em5"` This argument is used by the Control-M/Server security mechanism. Under certain circumstances, it cannot be modified. For more information, see the Security chapter and the description of the AuthorSecurity system parameter in GUI Server parameters.	
	FROM	String. Mandatory.
	TO	String. Optional.
FILE_PATH	Name of the library/directory in which the job script resides. String. Optional. `FILE_PATH FROM="File1" TO="File4"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CMDLINE	Command string supplied when the job Task Type is Command. Optional. `CMDLINE FROM="C:\Format" TO="C\:CD Emnt"`	
	FROM	String. Mandatory.
	TO	String. Optional.
HOSTID	Host name of an agent computer or a host group to which the job is submitted. Optional. `HOSTID FROM="Com3" TO="Acct4"`	
	FROM	String. Mandatory.
	TO	Host name of the agent computer on which the job copy is running Optional.
RUN_AS	Owner (user ID) associated with the job. This parameter is used by the Control-M/Server security mechanism. Optional. `OWNER FROM="emuser" TO="emhr"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MAXRERUN	Maximum number of reruns that can be performed for the job. Optional. `MAXRERUN FROM="1" TO="3"`	
	FROM	String. Mandatory.
	TO	String. Optional.
TIMEFROM	Earliest time for submitting the job. Optional. `TIMEFROM FROM="1430" TO="1450"`	
	FROM	String. Mandatory.
	TO	String. Optional.
TIMETO	Latest time for submitting the job. Optional. `TIMETO FROM="1600" TO="1620"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
DUE_OUT	Time that the job is expected to finish. Optional. `DUE_OUT FROM="1500" TO="1530"`	
	FROM	String. Mandatory.
	TO	String. Optional.
PRIORITY	Control-M job priority. Optional. `PRIORITY FROM="AA" TO="1A"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CRITICAL	Indicates whether the job is a critical-path job in Control-M. Optional. `CRITICAL FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
CYCLIC	Indicates whether the job is cyclic (to be run at regular intervals). Optional. `CYCLIC FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
CYCLIC_TYPE	Determines the type of cyclic job: ▪ Interval ▪ IntervalSequence ▪ SpecificTimes Relevant for Control-M version 6.4.01 or later.	
CYCLIC_ TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Relevant for Control-M version 6.4.01 or later.	
CYCLIC_INTERVAL_ SEQUENCE	A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas. Relevant for Control-M version 6.4.01 or later.	

Parameter	Description	
CYCLIC_TIMES_ SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.	
CONFIRM	Indicates whether the job must be manually confirmed by the Control-M/EM user before it runs. Optional. `CONFIRM FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
AUTOARCH	Determines whether SYSDATA is to be archived. Optional. `AUTOARCH FROM=0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
INTERVAL	Length of time (in minutes) to wait between reruns or cyclic runs of a job. Integer. Optional. `INTERVAL FROM="3" TO="4"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
OVERRIDE_PATH	Name of an alternate job script library/directory. String. Optional. `OVERRIDE PATH FROM="lib3" TO="lib4"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MAXWAIT	Number of extra days (after the original scheduling date) that the job is allowed to remain in the Active Jobs database awaiting execution. Integer. Optional. `MAXWAIT FROM="4" TO="3"`	
	FROM	Integer. Mandatory.
	TO	Integer. Optional.
DESCRIPTION	Free text description of the job. String. Optional. `DESCRIPTION FROM="data backup from 120399" TO="data backup from 021400"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DOCMEM	Name of the file containing job documentation. String. Optional. `DOCMEM FROM="mem4" TO="Mem67"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DOCLIB	Name of library or directory containing the job documentation file. String. Optional. `DOCLIB FROM="AcctFiles" TO="HRFiles"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DAYS	Days of the month on which to order the job. String. Optional. `DAYS FROM="ALL" TO="159"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
DAYS_AND_OR	Relationship between specified Days values and Weekdays values. Optional. `DAYS_AND_OR FROM="AND" TO="OR"`	
	FROM	String. Mandatory.
	TO	String. Optional.
WEEKDAYS	Days of the week on which to order the job. String. Optional. `WEEKDAYS FROM="1,2,4" TO="ALL"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DATE	Specific dates on which to order the job. String. MMDD format. Optional. `DATE FROM="0312" TO="0319"`	
	FROM	String. Dates are written in mmdd format. Mandatory. There is no delimiter between dates. For example, January 10 is written: DATE="0110."
	TO	String. Dates are written in mmdd format. Optional. There is no delimiter between dates. For example, January 10 is written: DATE="0110."
DAYSCAL	User-defined calendar used to specify a set of days. String. Optional. `DAYSCAL FROM="shipping" TO="receiving"`	
	FROM	String. Mandatory.
	TO	String. Optional.
WEEKSCAL	Calendar to be used to validate specified weekdays on which to order the job. String. Optional. `WEEKSCAL FROM="w5" TO="w6"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the job. String. Optional. `CONFCAL FROM="cal99" TO="cal00"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RETRO	Indicates whether the job is scheduled for possible execution after its original scheduling date (odate) has passed. Optional. `RETRO FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
SHIFT	Describes how to shift the scheduling date of the job. Optional. `SHIFT FROM="PREVDAY" TO="NEXTDAY"`	
	FROM	Mandatory. Valid values: ▪ IGNOREJOB ▪ PREVDAY ▪ NEXTDAY ▪ NOCONFCAL
	TO	Optional. Valid values: Same as mandatory FROM values.
SHIFTNUM	Number of days to shift the scheduling date of the job. Optional. `SHIFTNUM FROM="-10" TO="5"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
MAXDAYS	Maximum number of days to retain the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of AUTOARCH. String. Optional. `MAXDAYS FROM="07" TO="14"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MAXRUNS	Maximum number of job runs to retain the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of AUTOARCH. String. Optional. `MAXRUNS FROM="100" TO="250"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RERUNMEM	(*z/OS only*) JCL member to use when the job is automatically rerun. String. 1 - 8 characters. Optional. `RERUNMEM FROM="Mem45" TO="Mem7"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RETEN_DAYS	(*z/OS only*) Number of days to retain the job in the History Jobs file. String. Optional. `RETEN_DAYS FROM="5" TO="7"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RETEN_GEN	(*z/OS only*) Maximum number of generations to keep in the History Jobs file. String. Optional. `RETEN_GEN FROM="3" TO="4"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
PREV_DAY	Optional. `PREV_DAY FROM="N" TO="Y"`	
	FROM	Mandatory. Valid values: ▪ Y ▪ N
	TO	Optional. Valid values: ▪ Y ▪ N
IND_CYCLIC	Indicates whether the time interval between runs of a cyclic job is counted from the start or the end of the previous job run. Optional. `IND_CYCLIC FROM="Y" TO="N"`	
	FROM	Mandatory. Valid values: ▪ START ▪ END
	TO	Optional. Valid values: ▪ START ▪ END
RULE_BASED_ CALENDAR_ RELATIONSHIP	Relationship (AND|OR) between the specified Rule-Based Calendar and the job's own basic scheduling criteria. Optional. `RULE_BASED_CALENDAR_RELATIONSHIP FROM="AND" TO="OR"`	
	FROM	Mandatory. Valid values: ▪ AND ▪ OR
	TO	Optional. Valid values: ▪ AND ▪ OR
TAG_ RELATIONSHIP	Relationship (AND|OR) between the specified Schedule Tag criteria and the job's own basic scheduling criteria. This parameter is relevant only for jobs in a SMART Folder. Optional. This parameter is for backward compatibility. `TAG_RELATIONSHIP FROM="AND" TO="OR"`	

Parameter	**Description**	
	FROM	Mandatory. Valid values: ■ AND ■ OR
	TO	Optional. Valid values: ■ AND ■ OR
SYSDB	Determines whether single or multiple data sets are used to catalogue sysdata. Optional. `SYSDB FROM="1" TO="0"`	
	FROM	Mandatory. Valid values: ■ 0 (Multiple. Default) ■ 1 (Single)
	TO	Optional. Valid values: ■ 0 (Multiple. Default) ■ 1 (Single)
PDSNAME	Name of partitioned dataset (PDS) to be checked for free space. String. Optional. `PDSNAME FROM="Lib_3" TO="Lib_5"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MINIMUM	Minimum number of free partitioned dataset tracks required by the library specified for the PDSNAME parameter. Integer. Optional. `MINIMUM FROM="5" TO="6"`	
	FROM	Integer. Mandatory.
	TO	Integer. Optional.

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="3"><em>CATEGORY</em></td><td colspan="2">Name of a Control-D report decollating mission category that must be scheduled under Control-D when the job is scheduled under Control-M. String. Optional.
<code>CATEGORY FROM="*" TO="DAILY"</code></td></tr>
<tr><td>FROM</td><td>String. Mandatory.</td></tr>
<tr><td>TO</td><td>String. Optional.</td></tr>
</table>

Parameter	**Description**	
PREVENTNCT2	(*z/OS only*) Prevents dataset cleanup before the original job run Optional. Valid values: ■ Blank – Does not perform data set cleanup before the original job run. Default. ■ N – Does not prevent cleanup. ■ Y - Prevents data set cleanup. This value is not valid for started tasks. ■ L (List) – Do not perform data set cleanup before the original job run. Do generate messages that would be required for CDG adjustment during restart. ■ F (Flush) – Halt processing of the job if any data set cleanup error is detected (even if z/OS would not have stopped processing the job). `PREVENTNC2 FROM="1" TO="0"`	
	FROM	Mandatory. Valid values: ■ 0 (Do not prevent) ■ 1 (Prevent)
	TO	Optional. Valid values: ■ 0 (Do not prevent. Default.) ■ 1 (Prevent)

*JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,
OCT, NOV, DEC*
	Months when the job can run. Optional. Not including a month is the same as including that month with the value 0.
	`JAN FROM="0" TO="1"`
	`JUL FROM="0" TO="1"`	

	FROM	Mandatory. Valid values: ■ 0 (Do not run the job. Default) ■ 1 (Run the job.)
	TO	Optional. Valid values: ■ 0 (Do not run the job. Default) ■ 1 (Run the job.)
OPTION	Job output (Output) handling options. Optional. `OPTION FROM="Copy" TO="Release"`	

Parameter	Description	
	FROM	Mandatory. Valid values: ▪ Release ▪ Delete ▪ Copy ▪ Move ▪ File ▪ NewDest ▪ ChangeClass
	TO	Optional. Valid values: same as mandatory FROM values.
PAR	Certain OPTION values require that you supply additional information (such as Release, NewDest). The PAR parameter holds that information as a string. Optional. `PAR FROM="mem3log" TO="mem5log"`	
	FROM	String. Mandatory.
	TO	String. Optional.
FROM	Limits the OUTPUT handling operation to OUTPUTs from the specified class. Optional. `FROM FROM="1" TO="2"`	
	FROM	String. Mandatory.
	TO	String. Optional.
ADJUST_COND	Indicates whether to ignore prerequisite conditions normally set by predecessor jobs if the relevant predecessor jobs are not scheduled. This parameter is relevant only for jobs in a groSMART Foldere. Optional. Valid values: ▪ 0 (Do not ignore. Default.) ▪ 1 (Ignore relevant prerequisite conditions.) `ADJUST_COND FROM="1" TO="0"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
APPL_TYPE	Type of external application (for example, SAP or Oracle) on which the external application job runs. Mandatory for external application jobs. `APPL_TYPE FROM="SAP" TO="OracleApps"`	
	FROM	Mandatory. String. Up to 10 characters.
	TO	Optional. String.
APPL_VER	Version of the external application (for example, SAP or Oracle) on which the external application job runs. Mandatory for external application jobs. `APPL_VER FROM="4.5" TO="4.6"`	
	FROM	Mandatory. String. Up to 10 characters.
	TO	Optional. String.
APPL_FORM	Predefined set of external application parameters that are displayed in the Job Properties team. Mandatory for external application jobs. `APPL_FORM FROM="Default SAP 4.6" TO="Default SAP 4.5"`	
	FROM	Mandatory. String. Up to 30 characters.
	TO	Optional. String.
CM_VER	Version of external Application Add-on (for example, SAP or Oracle) that is installed in the Control-M installation. Mandatory for external application jobs. `CM_VER FROM="6.1.00" TO="6.1.01"`	
	FROM	Mandatory. String. Up to 10 characters.
	TO	Optional. String.

Parameter	Description	
MULTY_AGENT	When selected, broadcasts job submission details to all agents in a specified Host Group. Optional. `MULTY_AGENT FROM="N" TO="Y"`	
	FROM	Mandatory. Valid values: ▪ Y – Run as multi-agent job ▪ N – Do not run as multi-agent job. Default.
	TO	Optional. String.
ACTIVE_FROM	(*z/OS only*) Start of a period of time during which the job or SMART Folder can be ordered. Optional. `ACTIVE_FROM FROM="20070315" TO="20070601"`	
	FROM	Mandatory. Date Format: YYYYMMDD
	TO	Optional. String.
ACTIVE_TILL	(*z/OS only*) End of a period of time during which the job or SMART Folder can be ordered. Optional. `ACTIVE_TILL FROM="20070315" TO="20070601"`	
	FROM	Mandatory. Date Format: YYYYMMDD
	TO	Optional. String.
TIMEZONE	Global time zone used to calculate the interval for time-related conditions. Optional. `TIMEZONE FROM="EST" TO="GMT"`	
	FROM	Mandatory. String. Default: GMT
	TO	Optional. String.
SYSTEM_AFFINITY	Identity of the system in which the job must be initiated and executed (in JES2). Identity of the processor on which the job must execute (in JES3). Optional. String.	
	FROM	String. Mandatory. `SYSTEM_AFFINITY FROM="SYS3"`

Parameter	Description	
	TO	String. Optional. SYSTEM_AFFINITY FROM="SYS3" TO="SYS6"
REQUEST_NJE_HOST	Host in the JES network on which the job is to execute.	
	FROM	String. Mandatory. REQUEST_NJE_HOST="OS5"
	TO	String. Optional. REQUEST_NJE_HOST FROM="OS5" TO="OS6"
SCHEDULING_ENVIRONMENT	(*z/OS only*) Indicates the JES2 workload management scheduling environment associated with the job.	
	FROM	String. Mandatory. SCHEDULING_ENVIRONMENT FROM="SCHD2"
	TO	String. Optional. SCHEDULING_ENVIRONMENT FROM="SCHD2" TO="SCHD3"
CREATION_USER	Name of the user that created the job. String. Optional. CREATION_USER FROM="emuser" TO="em1"	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_DATE	Date that the job was created. String. Optional. CREATION_DATE FROM="1212" TO="2012"	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_TIME	Time that the job was created. String. Optional. CREATION_TIME FROM="1230" TO="1430"	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CHANGE_USERID	Name of the user that last modified the job. String. Optional. `CHANGE_USERID FROM="emuser" TO="emacct"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CHANGE_DATE	Date that the job was last modified. String. Optional. `CHANGE_DATE FROM="1204" TO="1304"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CHANGE_TIME	Time that the job was last modified. String. Optional. `CHANGE_TIME FROM="1650" TO="1700"`	
	FROM	String. Mandatory.
	TO	String. Optional.

duplicatedefjob arguments file example

The following example argument files are used with the duplicatedefjob utility:

Copy and modify definitions

Copy job processing definitions from the Tbl5NYC SMART Folder that have FOLDER_DSN Lib1 and JOBNAME Job3. Change FOLDER_DSN to Lib1_COPY and change JOBNAME to Job3_COPY. Store the changed definitions in the same SMART Folder.

```
<DUPLICATEJOB>
<FOLDER_NAME FROM="Tbl5NYC"/>
<FOLDER_DSN FROM="Lib1" TO="Lib1_COPY"/>
<JOBNAME FROM="Job3" TO="Job3_COPY"/>
</DUPLICATEJOB>
```

Duplicate jobs based on several criteria

Copy all cyclic jobs in the GrpAcct group whose jobname begins with "**Acct**". Append "_COPY" to the job name of each copied job.

```
<DUPLICATEJOB>
<SUBAPPLICATION FROM="GrpAcct"/>
```

```
<CYCLIC FROM="1"/>
<JOBNAME FROM="Acct*" TO="Acct*_COPY"/>
</DUPLICATEJOB>
```

exportdefjob

The exportdefjob utility exports job processing definitions from a folder in the Control-M/EM database to an output file. To run the utiltity, see Running the exportdefjob utility (on page 110).

When exportdefjob is invoked, it processes a specified file of arguments in XML format. This file contains statements that identify existing job processing definitions. The identified definitions are exported from the Control-M/EM database to an output file. You can modify the exported job processing definitions in the output file and can import the modified definitions into the Control-M/EM database using either the defjob or updatedef utility. For more information, see exportdefjob arguments file (on page 112).

Running the exportdefjob utility

This procedure describes how to run the exportdefjob utiltiy, which enables you to export job processing definitions from a folder in the Control-M/EM database to an output file.

➢ To run the exportdefjob utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter one of the following commands:

- `emdef exportdefjob [-USERNAME <user> [-PASSWORD <password>] | -PASSWORD_FILE <password file>] -HOST <Gui Server Name> -ARG_FILE <arg file name> -OUT_FILE <out file name>`
- `emdef exportdefjob [-u <user> [-p <password>] | -pf <password file>] -s <Gui Server Name> -arg <arg file name> -out <out file name>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the exportdefjob parameters and switches, see exportdefjob parameters (on page 111) and exportdefjob switches (on page 112) .

exportdefjob parameters

The following table describes the exportdefjob utility parameters:

Parameter	Description
<user name>	Control-M for Databases user name.
<password>	The Control-M for Databases user password.

<password file name>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

<GUI Server Name>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<XML file name>	Path and name of the arguments file containing exportdefjob specifications. For information about this file, see XML file preparation (on page 616).
<out file name>	Path and name of the file containing the exported job specifications.
CYCLIC_ TYPE	Determines the type of cyclic job: ▪ Interval ▪ IntervalSequence ▪ SpecificTimes Relevant for Control-M version 6.4.01 or later.
CYCLIC_ TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Relevant for Control-M version 6.4.01 or later.
CYCLIC_ INTERVAL_ SEQUENCE	A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas. Relevant for Control-M version 6.4.01 or later.
CYCLIC_ TIMES_ SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.
-ctm	Name of the Control-M installation that processes the job.

Parameter	Description
-folder	Name of the folder.
-app	Name of the application to which the job's group belongs.
-subapp	Name of the group to which the job belongs.

exportdefjob switches

The following table describes an optional switch for the duplicatedefjob utility:

Switch	Description
?	Displays utility's description and available options.
/a	The /a switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The /a switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to 2 or 3.
/v	Used to receive verbose messages.
/t	Operate on a single folder at a time, to reduce process memory.

exportdefjob arguments file

The following rules apply to the exportdefjob argument file:

- More than one job can be specified in an exportdefjob file.
- The arguments file is case-sensitive.
- All parameter values must be enclosed in quotation marks (for example, JOBNAME="Job1").
- More than one PARAM parameter can be used in a TERM statement.
- The relationship between PARAM parameters in a TERM statement is AND.

The exportdefjob arguments file is checked and processed. If there are any errors, a message is displayed specifying the lines with the errors. The exported job processing definitions are saved to the output file whose name and location is specified in the **-out** *outFileName* parameter.

For more information on the arguments file parameters for the exportdefjob utility, see exportdefjob arguments file parameters (on page 113) , and exportdefjob arguments file sample (on page 114).

exportdefjob arguments file parameters

The following table describes exportdefjob arguments file parameters:

Parameter	Description	
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the .dtd file.		
TERMS	These tags indicate the start and end of the TERMS file. Only criteria that are located between the tags are considered to be part of the argument.	
TERM	The TERM tags indicate the start and the end of a group of selection criteria used to specify a job or jobs that are to be exported. Only PARAM tags that are located between the TERM tags are considered to be part of the TERM argument.	
	REL	Relationship between terms. Optional. Valid values: ▪ AND ▪ OR
PARAM	Selection criteria parameter used to determine the job definitions that are to be exported. More than one PARAM can be specified. Mandatory. `PARAM NAME="DATACENTER" OP="E" VALUE="Center1"`	
	NAME	String. Mandatory. The parameter name of any job processing definition parameter. At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, OLDER_NAME, FOLDER_DSN
	OP	Relationship between the NAME and VALUE parameters of the TERM. Mandatory. Valid values: ▪ EQ – equal ▪ NEQ – not equal ▪ LIKE – mask or pattern
	VALUE	String. Mandatory. Value of the parameter specified in the NAME field.

exportdefjob arguments file sample

The following example argument files are used with the exportdefjob utility:

Export job definitions based on one or more criteria

Export job processing definitions that either:

have data center name Data1 and a jobname that begins with the letter **J.**

- or -

have jobname Job5 and are not cyclic jobs.

```
<TERMS>
<TERM>
<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>
<PARAM NAME="JOBNAME" OP="LIKE" VALUE="J*"/>
</TERM>
<TERM>
<PARAM NAME="JOBNAME" OP="EQ" VALUE="Job5"/>
<PARAM NAME="CYCLIC" OP="EQ" VALUE="0"/>
</TERM>
</TERMS>
```

Export based on multiple criteria

Export all job processing definitions from either the **Data1** or **Data2** data center that have a JobName that does not begin with the letter **R.**

```
<TERMS>
<TERM>
<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>
<PARAM NAME="JOBNAME" OP="NEQ" VALUE="R*"/>
</TERM>
<TERM>
<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data2"/>
<PARAM NAME="JOBNAME" OP="NEQ" VALUE="R*"/>
</TERM>
</TERM>
</TERMS>
```

loopdetecttool

The loopdetecttool utility checks job processing definitions to determine if conditions are defined in a way that would cause loops. A loop in this context means:

- a chain of jobs that will never run because the IN condition needed to start the first job in the chain will be created only by the last job in the chain
- any combination of jobs, groups, and conditions between them, because it is not clear which job in the group will run first or last, creating or deleting the relevant conditions

To run the utility, see Running the loopdetecttool utility (on page 115).

The loopdetecttool parameter of the emdef utility reads an argument file (in XML format) that contains criteria that determine which job processing definitions and SMART Folder the utility should analyze. The utility checks for definitions whose conditions could potentially cause a loop. for more information, see loopdetecttool arguments file (on page 117).

Running the loopdetecttool utility

The following procedure describes how to run the loopdetecttool utility, which enables you to check job processing definitions to determine if conditions are defined in a way that would cause loops.

➢ To run the loopdetecttool utility:

1. Perform one of the following actions:
   - For UNIX:

     Log on to a Control-M/Enterprise Manager account.
   - For Windows:

     Open a command prompt window on a computer on which Control-M/EM is installed.

     For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the appropriate command:
   - For UNIX:

     ```
     emdef loopdetecttool [-U <user> [-P <password>] | -pf <password file>]
     -s <GUI Server Name> -arg <arg file name> -out <out file name>
     ```
   - For Windows:

     ```
     emdef loopdetecttool [-U <user> [-P <password>] | -pf <password file>]
     -s <GUI Server Name> -arg <arg file name> -out <out file name>
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the loopdetecttool parameters, see loopdetecttool parameters (on page 116).

loopdetecttool parameters

The following table describes the loopdetecttool parameters:

Parameter	Description
<user name>	Control-M for Databases user name
<password>	Control-M for Databases user password

<password file>	Flat file that contains an unencrypted user name and password on separate lines in the following format:
	user=<*userName*>
	password=<*password*>

<GUI Server Name>	Control-M/EM GUI server logical name
<arg file name>	Path and name of the arguments file that contains criteria for job and SMART Folder specifications. For information about this file, see the Arguments file rules section below and XML file preparation (on page 616).
<out file name>	Path and name of the output file that contains the summary of problematic jobs and conditions (loops).

loopdetecttool arguments file

The arguments file is processed. Corresponding definitions in the Control-M/EM database are checked. A summary that lists problematic definitions and conditions (loops) is produced in the output file whose name and location is specified in the *outFileName* parameter.

Arguments file rules

The following rules apply to the loopdetecttool argument file:

- The arguments file is case-sensitive.
- More than one PARAM parameter can be used in a TERM statement.
- The relationship between PARAM parameters in a TERM statement is AND.
- The default relationship between TERM statements is OR.
- All parameter values must be enclosed in quotation marks (for example, JOBNAME="Job1").

The output file is in XML format and is structured as follows:

- All loops are listed between ctmem:loops tags.
- For each loop found, problematic jobs and conditions are listed.

For more information on the arguments file parameters for the loopdetecttool utility, seeloopdetecttool arguments file parameters (on page 117).

loopdetecttool arguments file parameters

The following table describes arguments file parameters for the loopdetecttool utiltity:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td colspan="3">The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the <b>.dtd</b> file. These lines must appear exactly as follows:
?xml version=’1.0’ encoding=’UTF-8’?
!DOCTYPE TERMS SYSTEM "terms.dtd"</td></tr>
<tr><td><i>TERMS</i></td><td colspan="2">These tags indicate the start and end of the TERMS file. Only criteria that are located between the tags are considered part of the argument.</td></tr>
<tr><td rowspan="2"><i>TERM</i></td><td colspan="2">The TERM tags indicate the start and the end of a group of selection criteria used to specify a job or jobs that are to be analyzed. Only PARAM tags that are located between the TERM tags are considered part of the TERM argument.</td></tr>
<tr><td>REL</td><td>The relationship between the terms (optional). Valid values are:
▪ AND
▪ OR</td></tr>
</table>

Parameter	Description	
PARAM	Selection criteria parameter used to determine the job definitions that are to be analyzed. You can specify more than one PARAM. This parameter is required. `PARAM NAME="DATACENTER" OP="EQ" VALUE="Center1"`	
	NAME	The parameter name of any job processing definition parameter. This parameter is required. At least one of the following SMART Folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN
	OP	The relationship between the NAME and VALUE parameters of the TERM. This relationship is required. Valid values are: ▪ EQ – equal ▪ NEQ – not equal ▪ NOTIN – does not contain ▪ LIKE – mask or pattern
	VALUE	The value of the parameter specified in the NAME field. This value is required.

loopdetecttool output example

The following are examples of the loopdetecttool output:

output file example:

```
<!DOCTYPE ctmem:loop_cond_detect SYSTEM "C:\Program Files\Altova\XML
Spy Suite\Examples\LoopDetectOut.dtd">
<ctmem:loop_cond_detect>
  <ctmem:message>1 Loop was detected.</ctmem:message>
  <ctmem:loops>
    <ctmem:message>2 Jobs were found in the loop</ctmem:message>
    <ctmem:loop>
      <ctmem:job>
        <ctmem:control_m>PROD_DC1</ctmem:control_m>
        <ctmem:order_folder>Daily_Prod1</ctmem:order_folder>
        <ctmem:application>WinDcProd1</ctmem:application>
        <ctmem:group>BackupDaily</ctmem:group>
        <ctmem:job_name>Job1</ctmem:job_name>
      </ctmem:job>
      <ctmem:condition>
        <ctmem:cond_name>Job1-Ended-OK</ctmem:cond_name>
        <ctmem:cond_date>ODAT</ctmem:cond_date>
        <ctmem:message>regular condition</ctmem:message>
      </ctmem:condition>
      <ctmem:job>
        <ctmem:control_m>PROD_DC1</ctmem:control_m>
        <ctmem:order_folder>Daily_Prod1</ctmem:order_folder>
        <ctmem:application>WinDcProd1</ctmem:application>
        <ctmem:group>BackupDaily</ctmem:group>
        <ctmem:job_name>Job2</ctmem:job_name>
      </ctmem:job>
      <ctmem:condition>
        <ctmem:cond_name>Job2-Ended-OK</ctmem:cond_name>
        <ctmem:cond_date>ODAT</ctmem:cond_date>
        <ctmem:message>regular condition</ctmem:message>
      </ctmem:condition>
    </ctmem:loop>
  </ctmem:loops>
</ctmem:loop_cond_detect>
```

Analyze job definitions for loops based on one or more criteria

The following example file specifies to analyze job processing definitions that have either of the following conditions:

- The data center name is Data1, and a job name begins with the letter **J**.
- The job name is Job5, and the job is not cyclic.

```
<?xml version=’1.0’ encoding=’UTF-8’?>
<!DOCTYPE TERMS SYSTEM "terms.dtd">
<TERMS>
  <TERM>
    <PARAM NAME="DATACENTER" OP="EQ" VALUE="Data1"/>
    <PARAM NAME="JOBNAME" OP="LIKE" VALUE="J*"/>
  </TERM>
  <TERM>
    <PARAM NAME="JOBNAME" OP="EQ" VALUE="Job5"/>
    <PARAM NAME="CYCLIC" OP="EQ" VALUE="0"/>
  </TERM>
</TERMS>
```

Analyze job definitions for loops based on multiple criteria

The following example file specifies to analyze all job processing definitions from either the **Data1** or **Data2** data center that have a job name that does not begin with the letter **R.**

```
<?xml version=’1.0’ encoding=’UTF-8’?>
<!DOCTYPE TERMS SYSTEM "terms.dtd">
<TERMS>
  <TERM>
    <PARAM NAME="DATACENTER" OP="EQ" VALUE="Data1"/>
    <PARAM NAME="JOBNAME" OP="NEQ" VALUE="R*"/>
  </TERM>
  <TERM>
    <PARAM NAME="DATACENTER" OP="EQ" VALUE="Data2"/>
    <PARAM NAME="JOBNAME" OP="NEQ" VALUE="R*"/>
  </TERM>
</TERMS>
```

loopdetecttool output

The following table lists the summary of the problematic jobs and conditions in an XML format file after the utility runs.

Parameter	Description
The beginning of the arguments file specifies the location of the **.dtd** file as follows: !DOCTYPE "ctmem:loop_cond_detect" SYSTEM "path\filename.dtd"	
ctmem:message	Between these tags, relevant messages are listed, such as the number of loops detected, how many jobs were found in each loop, and other remarks.
ctmem:loops	Between these tags, details about all found loops are listed.
ctmem:loop	Between these tags, details about each found loop is listed. Each ctmem:loop tag contains pairs of the following tags: ctmem:job and ctmem:condition.
ctmem:job	Between these tags, information about the job that is part of a potential loop is displayed. This information includes the job's data center, SMART Folder, group, application, and job name. Pairs of jobs and conditions are grouped by In and Out conditions.
ctmem:condition	Between these tags, information about the conditions that were found to cause a potential loop is displayed. This information includes the condition's name, date, and type. Pairs of jobs and conditions are grouped by In and Out conditions.

updatedef

The updatedef utility updates (modifies) specified parameter values in the following definitions in the Control-M/EM database:

- Job processing definitions
- Folder definitions
- SMART Folder definitions
- Sub-folder definitions
- updatedef modifies the characteristics of existing job processing definitions.
- duplicatedefjob creates new job processing definitions based on existing job processing definitions in the "from" data center and folders.

To run the updatedef utility, see Running the updatedef utility (on page 122).

The selected jobs, folders, SMART Folders and Sub-folders are modified according to specifications in the updatedef arguments file. The updatedef utility does not create new jobs or folders. For more information, see updatedef arguments file (on page 124).

The updatedef arguments file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

Running the updatedef utility

This procedure describes how to run the updatedef utility, which enables you to update specified parameter values in the Control-M/EM database.

➢ To run the updatedef utility:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter either of the following commands:

   -
     ```
     emdef updatedef [-u <user name> [-p <password>] | -pf <password file>]
     -s <GUI Server Name> -arg <XML file name> [/a]
     ```

   -
     ```
     emdef updatedef [-USERNAME <user name> [-PASSWORD <password>] |
     -PASSWORD_FILE <password file>] -HOST <GUI Server Name> -ARG_FILE <XML
     file name> [/a]
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the updatedef parameters and switches, see updatedef parameters (on page 123) and updatedef switches (on page 123).

updatedef parameters

The following table lists the updatedef utility parameters:

Parameter	Description
<*user name*>	Control-M for Databases user name.
<*password*>	The Control-M for Databases user password.

<*password file*>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

<*GUI Server Name*>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<*XML file name*>	Path and name of the arguments file containing updatedef specifications. For information about preparing this file, see XML file preparation (on page 616).

updatedef switches

The following table lists an optional switch for the duplicatedefjob utility:

Switch	Description
?	Displays utility's description and available options.
/a	The /a switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The /a switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to 2 or 3.
/v	Used to receive verbose messages.

updatedef arguments file

The following rules apply to the updatedef arguments file:

- Only one block of criteria can be specified for updating in each arguments file. This criteria block can update many jobs or folders.
- Text in the arguments file is case-sensitive.
- All parameter values must be enclosed in quotation marks (for example, JOBNAME="Job1").

Multiple values can be specified by using the * wildcard character. For an explanation of how wildcards function in the XML-based utilities, see Abbreviations and conventions (on page 10)

Most parameters of the job, folder, SMART Folder or Sub-folder definitions are optional. However:

- If you specify a parameter, its FROM subparameter is mandatory and its TO subparameter is optional.
- When a FROM value is specified without a TO value, the FROM value is used as a filter criterion.
- When a TO value is included, it indicates the new value to which the parameter is set.
- SMART Folder parameters are usually modified using the SMART Folder criteria described in Arguments file parameters for SMART folders (on page 146). However, you cannot use these criteria to change the name of a SMART Folder.
- The folder name parameter of a job processing definition cannot be modified using the job definition criteria described in Arguments file parameters for jobs (on page 127). However, you can use the **updatedef** utility to change the name of a SMART Folder or folder by using the criteria for folders (described in Arguments file parameters for folders (on page 125)) and specifying a new value for the FOLDER_NAME parameter.

Three sets of parameters can be supplied in an arguments file – one each for folders, SMART Folders, and jobs. Each set of parameters is described in:

- Arguments file parameters for folders (on page 125)
- Arguments file parameters for SMART folders (on page 146)
- Arguments file parameters for jobs (on page 127)
- loopdetecttool output example (on page 118).

Arguments file parameters for folders

The following table lists the arguments file parameters for folders:

Parameter	Description	
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the .dtd file.		
UPDATE	These tags indicate the start and end of the UPDATE argument. Only criteria located between the tags are considered to be part of the argument. Mandatory.	
FOLDER	These tags indicate the start and end of the folder specification. Criteria identifying the folders to be modified and indicating the types of modifications to be made are located between these tags. Optional. If you are using Control-M version 6.4.00 or earlier the parameter SCHED_FOLDER will be used in place of FOLDER.	
FOLDER_NAME	Name of the folder to which the job belongs. Optional. `FOLDER_NAME FROM="Tbl5NYC" TO="Tbl_new"`	
	FROM	String. Mandatory.
	TO	String. Optional.
FOLDER_DSN	(*z/OS only*) Name of the library that contains the (scheduling) folder. Optional. *ABLE_DSN FROM="Lib1" TO="Lib2"*	
	FROM	String. Mandatory.
	TO	String. Optional.
DATACENTER	Name of the Control-M installation to which the job belongs. Optional. A TO parameter cannot be specified for this parameter. `DATACENTER FROM="CTMNYC"`	
	FROM	String. Mandatory.
FOLDER_ORDER_ METHOD	Optional. `FOLDER_ORDER_ METHOD FROM="Job3A"`	
	FROM	String. Mandatory.

Parameter	**Description**	
	TO	String. Optional
CYCLIC_TYPE	Determines the type of cyclic job: - Interval - IntervalSequence - SpecificTimes Relevant for Control-M version 6.4.01 or later.	
CYCLIC_ TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Relevant for Control-M version 6.4.01 or later.	
CYCLIC_INTERVAL_ SEQUENCE	A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas. Relevant for Control-M version 6.4.01 or later.	
CYCLIC_TIMES_ SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.	

Arguments file parameters for jobs

The following table lists the arguments file parameters for jobs:

Parameter	Description	
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the .dtd file.		
UPDATE	These tags indicate the start and end of the UPDATE argument. Only criteria that are located between the tags are considered to be part of the argument. Mandatory.	
JOB	These tags indicate the start and end of each job specification. Criteria identifying the jobs to be modified and indicating the types of modifications to be made are located between these tags. Optional.	
FOLDER_NAME	Name of the folder to which the job belongs. Mandatory. This parameter cannot be modified. `FOLDER_NAME FROM="Tbl5NYC"`	
	FROM	String. Mandatory.
FOLDER_DSN	(*z/OS only*) Name of the library that contains the (scheduling) folder. Mandatory. This parameter cannot be modified. `FOLDER_DSN FROM="Lib1"`	
	FROM	String. Mandatory.
DATACENTER	Name of the Control-M installation to which the job belongs. Mandatory. This parameter cannot be modified. `DATACENTER FROM="CTMNYC"`	
	FROM	String. Mandatory.
FOLDER_ORDER_ METHOD	Optional. This parameter cannot be modified. `FOLDER_ORDER_ METHOD FROM="Job3"`	
	FROM	String. Mandatory.
JOBNAME	Name of the job processing definition. Optional. `JOBNAME FROM="Job3" TO="Job3_COPY"`	
	FROM	Mandatory.

Parameter	Description	
	TO	Optional.
FILE_NAME	Name of the file that contains the job script. Optional. `FILE_NAME FROM="File3" TO="File7"`	
	FROM	Mandatory.
	TO	Optional.
SUB_APPLICATION	Name of the group to which the job belongs. Optional. `GROUP FROM="Grp_HR"`	
	FROM	Mandatory.
APPLICATION	Name of the application to which the job’s group belongs. Optional. `APPLICATION FROM="App3" TO="App1"`	
	FROM	String. Mandatory.
	TO	String Optional.

Parameter	Description	
TASKTYPE	Type of the job (task) to be performed by Control-M. Optional. `TASKTYPE FROM="Detached" TO="Dummy"`	
	FROM	Mandatory. Valid values: - Job - Detached - Command - Dummy - External (*z/OS only*) Valid values: - Job - Started_Task In Control-M/EM versions earlier than 6.1.00, the TASKTYPE format contained: - for z/OS, emergency and cyclic information - for other operating systems, critical and cyclic information Control-M/EM version 6.2.01 and higher can run jobs with the old TASKTYPE format. However, BMC recommends that, to specify this type of information when creating new job processing definitions, you use CRITICAL and CYCLIC parameters. [for z/OS] BMC recommends that, to specify this type of information when creating new job processing definitions, you use CRITICAL (a value of "1" indicates that the job is an Emergency job) and CYCLIC parameters. Critical path jobs are indicated by coding an * as the first character in the Priority parameter. There is no connection between critical path jobs and the Critical parameter.
	TO	Optional. Valid values: Same as mandatory FROM values. (*z/OS only*) Valid values: Same as z/OS FROM values.

Parameter	Description	
CREATED_BY	Control-M/EM user who defined the job. This parameter is used by the Control-M/Server security mechanism. String. Optional. This argument is used by the Control-M/Server security mechanism and, under certain circumstances, cannot be modified. For more information, see the Security chapter and the description of the **AuthorSecurity** system parameter in GUI Server parameters. `CREATED_BY FROM="emuser" TO="em5"`	
	FROM	String. Mandatory.
	TO	String. Optional.
FILE_PATH	Name of the library/directory in which the job script resides. String. Optional. `FILE_PATH FROM="File1" TO="File4"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CMDLINE	Command string supplied when the job Task Type is Command. Optional. `CMDLINE FROM="C:\Format" TO="C\:CD Emnt"`	
	FROM	String. Mandatory.
	TO	String. Optional.
HOSTID	Host name of an agent computer or host group to which the job is submitted. Optional. `HOSTID FROM="Com3" TO="Acct4"`	
	FROM	String. Mandatory.
	TO	Host name of the agent computer on which the job copy is running Optional.
RUN_AS	Owner (user ID) associated with the job. This parameter is used by the Control-M/Server security mechanism. Optional. `OWNER FROM="emuser" TO="emhr"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	**Description**	
MAXRERUN	Maximum number of reruns that can be performed for the job. Optional. `MAXRERUN FROM="1" TO="3"`	
	FROM	String. Mandatory.
	TO	String. Optional.
TIMEFROM	Earliest time for submitting the job. Optional. `TIMEFROM FROM="1430" TO="1450"`	
	FROM	String. Mandatory.
	TO	String. Optional.
TIMETO	Latest time for submitting the job. Optional. `TIMETO FROM="1600" TO="1620"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DUE_OUT	Time that the job is expected to finish. Optional. `DUE_OUT FROM="1500" TO="1530"`	
	FROM	String. Mandatory.
	TO	String. Optional.
PRIORITY	Indicates Control-M job priority. Optional. `PRIORITY FROM="AA" TO="1A"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CRITICAL	Indicates whether the job is a critical-path job in Control-M. Optional. `CRITICAL FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
CYCLIC	Indicates whether the job is cyclic (to be run at regular intervals). Optional. `CYCLIC FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
	TO	Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)
CONFIRM	Indicates that the job must be manually confirmed by the Control-M/EM user before it runs. Optional. `CONFIRM FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (Default) ▪ 1
	TO	Optional. Valid values: ▪ 0 (Default) ▪ 1
AUTOARCH	Determines whether SYSDATA is to be archived. Optional. `AUTOARCH FROM=0" TO="1"`	

Parameter	Description	
	FROM	Mandatory. Valid values: ■ 0 (No. Default) ■ 1 (Yes)
	TO	Optional. Valid values: ■ 0 (No. Default) ■ 1 (Yes)
INTERVAL	Length of time (in minutes) to wait between reruns of a job or between cyclic runs of a job. Integer. Optional. `INTERVAL FROM="3" TO="4"`	
	FROM	String. Mandatory.
	TO	String. Optional.
OVERRIDE_PATH	Name of an alternate job script library/directory. String. Optional. `OVERRIDE PATH FROM="lib3" TO="lib4"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MAXWAIT	Number of extra days (after the original scheduling date) that the job is allowed to remain in the Active Jobs database awaiting execution. Integer. Optional. `MAXWAIT FROM="4" TO="3"`	
	FROM	Integer. Mandatory.
	TO	Integer. Optional.
DESCRIPTION	Free text description of the job. String. Optional. `DESCRIPTION FROM="data backup from 120399" TO="data backup from 021400"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
DOCMEM	Name of the file containing job documentation. String. Optional. `DOCMEM FROM="mem4" TO="Mem67"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DOCLIB	Name of a library or directory containing the job documentation file. String. Optional. `DOCLIB FROM="AcctFiles" TO="HRFiles"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DAYS	Days of the month on which to order the job. String. Optional. `DAYS FROM="ALL" TO="159"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DAYS_AND_OR	Indicates the relationship between specified Days values and Weekdays values. Optional. `DAYS_AND_OR FROM="AND" TO="OR"`	
	FROM	String. Mandatory.
	TO	String. Optional.
WEEKDAYS	Days of the week on which to order the job. String. Optional. `WEEKDAYS FROM="1,2,4" TO="ALL"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
DATE	Specific dates on which to order the job. String. mmdd format. Optional. `DATE FROM="0312" TO="0319"`	
	FROM	String. Mandatory. Dates are written in mmdd format. There is no delimiter between dates. For example, January 10 is written in this manner: DATE="0110"
	TO	String. Optional. Dates are written in mmdd format. There is no delimiter between dates. For example, January 10 is written in this manner: DATE="0110"
DAYSCAL	Name of a user-defined calendar used to specify a set of days. String. Optional. `DAYSCAL FROM="shipping" TO="receiving"`	
	FROM	String. Mandatory.
	TO	String. Optional.
WEEKSCAL	Name of a calendar to be used to validate specified weekdays on which to order the job. String. Optional. `WEEKSCAL FROM="w5" TO="w6"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the job. String. Optional. `CONFCAL FROM="cal99" TO="cal00"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
RETRO	Indicates whether the job is scheduled for possible execution after its original scheduling date (odate) has passed. Optional. `RETRO FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ■ 0 (No. Default) ■ 1 (Yes)
	TO	Optional. Valid values: ■ 0 (No. Default) ■ 1 (Yes)
SHIFT	Describes how to shift the scheduling date of the job. Optional. `SHIFT FROM="PREVDAY" TO="NEXTDAY"`	
	FROM	Mandatory. Valid values: ■ IGNOREJOB ■ PREVDAY ■ NEXTDAY ■ NOCONFCAL
	TO	Optional. Valid values: Same values as mandatory FROM.
SHIFTNUM	Number of days to shift the scheduling date of the job. Optional. `SHIFTNUM FROM="-10" TO="5"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MAXDAYS	Maximum number of days to retains the SYSDATA archive dataset for jobs that ended **NOTOK**. Subparameter of **AUTOARCH**. String. Optional. `MAXDAYS FROM="07" TO="14"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
MAXRUNS	Maximum number of job runs to retains the SYSDATA archive dataset for jobs that ended NOTOK. Subparameter of **AUTOARCH**. String. Optional. `MAXRUNS FROM="100" TO="250"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RERUNMEM	Name of the JCL member to use when the job is automatically rerun. String. 1 - 8 characters. Optional. `RERUNMEM FROM="Mem45" TO="Mem7"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RETEN_DAYS	(*z/OS only*) Number of days to retain the job in the History Jobs file. String. Optional. `RETEN_DAYS FROM="5" TO="7"`	
	FROM	String. Mandatory.
	TO	String. Optional.
RETEN_GEN	(*z/OS only*) Maximum number of generations of the job to keep in the History Jobs file. String. Optional. `RETEN_GEN FROM="3" TO="4"`	
	FROM	String. Mandatory.
	TO	String. Optional.
TASK_CLASS	Optional. `TASK_CLASS FROM="Distribution" TO="Decollation"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
PREV_DAY	Optional. PREV_DAY FROM="N" TO="Y"	
	FROM	Mandatory. Valid values: ▪ Y ▪ N
	TO	Optional. Valid values: ▪ Y ▪ N
IND_CYCLIC	Indicates whether the interval between further runs of a cyclic job is counted from the start or the end of the previous job run. Optional. IND_CYCLIC FROM="Y" TO="N"	
	FROM	Mandatory. Valid values: ▪ START ▪ END
	TO	Optional. Valid values: ▪ START ▪ END
RULE_BASED_ CALENDAR *_RELATIONSHIP*	Relationship (AND|OR) between the specified Rule-Based Calendar and the job's own basic scheduling criteria. Optional. RULE_BASED_CALENDAR_RELATIONSHIP FROM="AND" TO="OR"	
	FROM	Mandatory. Valid values: ▪ AND ▪ OR
	TO	Optional. Valid values: ▪ AND ▪ OR
TAG_RELATIONSHIP	Relationship (AND|OR) between the specified Schedule Tag criteria and the job's own basic scheduling criteria. This parameter is relevant only for jobs in a SMART Folder. Optional. This parameter is for backward compatibility. TAG_RELATIONSHIP FROM="AND" TO="OR"	

Parameter	Description	
	FROM	Mandatory. Valid values: ■ AND ■ OR
	TO	Optional. Valid values: ■ AND ■ OR
SYSDB	Determines whether one or multiple data sets are used to catalogue sysdata. Optional. `SYSDB FROM="1" TO="0"`	
	FROM	Mandatory. Valid values: ■ 0 (Multiple-Default) ■ 1 (Single)
	TO	Optional. Valid values: ■ 0 (Multiple-Default) ■ 1 (Single)
PDSNAME	Name of a partitioned dataset (PDS) to be checked for free space. String. Optional. `PDSNAME FROM="Lib_3" TO="Lib_5"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MINIMUM	Minimum number of free partitioned dataset tracks required by the library specified for the PDSNAME parameter. Integer. Optional. `MINIMUM FROM="5" TO="6"`	
	FROM	Integer. Mandatory.
	TO	Integer. Optional.

Parameter	Description	
CATEGORY	Name of a Control-D report decollating mission category that must be scheduled under Control-D when the job is scheduled under Control-M. String. Optional. `CATEGORY FROM="*" TO="DAILY"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
PREVENTNC2	(*z/OS only*) Performs dataset cleanup before the original job run. Optional. `PREVENTNC2 FROM="1" TO="0"`	
	FROM	Mandatory. Valid values: ■ 0 (Default) ■ 1
	TO	Optional. Valid values: ■ 0 (Default) ■ 1
JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC	Months when the job can run. Optional. Not including a month is the same as including a month with value 0. `JAN FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ■ 0 (Default) ■ 1
	TO	Optional. Valid values: ■ 0 (Default) ■ 1
OPTION	Job output (output) handling options. Optional. `OPTION FROM="Copy" TO="Release"`	
	FROM	Mandatory. Valid values: ■ Release ■ Delete ■ Copy ■ Move ■ File ■ NewDest ■ ChangeClass
	TO	Optional. Valid values: Same as mandatory FROM.

Parameter	Description	
PAR	Certain OPTION values require that you supply additional information (such as Release, NewDest). The PAR parameter holds that information as a string. Optional. `PAR FROM="mem3log" TO="mem5log"`	
	FROM	String. Mandatory.
	TO	String. Optional.
FROM	Limits the OUTPUT handling operation to OUTPUTs from the specified class. Optional. `FROM FROM="1" TO="2"`	
	FROM	String. Mandatory.
	TO	String. Optional.
ADJUST_COND	Indicates whether to ignore prerequisite conditions normally set by predecessor jobs if the relevant predecessor jobs are not scheduled. This parameter is relevant only for jobs in a SMART Folder. Optional. Valid values: ▪ 0 (Do not ignore. Default.) ▪ 1 (Ignore relevant prerequisite conditions) `ADJUST_COND FROM="1" TO="0"`	
	FROM	String. Mandatory.
	TO	String. Optional.
APPL_TYPE	Type of external application (for example, SAP or Oracle) on which the external application job runs. Mandatory for external application jobs. `APPL_TYPE FROM="SAP" TO="OracleApps"`	
	FROM	Mandatory. String. Up to 10 characters.
	TO	Optional. String.
APPL_VER	Version of the external application (for example, SAP or Oracle) on which the external application job runs. Mandatory for external application jobs. `APPL_VER FROM="4.5" TO="4.6"`	
	FROM	Mandatory. String. Up to 10 characters.
	TO	Optional. String.

Parameter	Description	
APPL_FORM	Predefined set of external application parameters that are displayed in the Job Properties team. Mandatory for external application jobs. `APPL_FORM FROM="Default SAP 4.6" TO="Default SAP 4.5"`	
	FROM	Mandatory. String. Up to 30 characters.
	TO	Optional. String.
CM_VER	Version of external Application Add-on(for example, SAP or Oracle) that is installed in the Control-M installation. Mandatory for external application jobs. `CM_VER FROM="6.1.00" TO="6.1.01"`	
	FROM	Mandatory. String. Up to 10 characters.
	TO	Optional. String.
MULTY_AGENT	When selected, broadcasts job submission details to all agents in a specified Host Group. Optional. `MULTY_AGENT FROM="N" TO="Y"`	
	FROM	Mandatory. Valid values: ▪ Y – run as multi-agent job. ▪ N – do not run as multi-agent job. Default.
	TO	Optional. String.
ACTIVE_FROM	(*z/OS only*) Start of a period of time during which the job or SMART Folder can be ordered. Optional. `ACTIVE_FROM FROM="20080315" TO="20080601"`	
	FROM	Mandatory. Date Format: YYYYMMDD
	TO	Optional. String.
ACTIVE_TILL	(*z/OS only*) End of a period of time during which the job or SMART Folder can be ordered. Optional. `ACTIVE_TILL FROM="20080315" TO="20080601"`	
	FROM	Mandatory. Date Format: YYYYMMDD
	TO	Optional. String.

Parameter	Description	
TIMEZONE	Global time zone used to calculate the interval for time-related conditions. Optional. `TIMEZONE FROM="EST" TO="GMT"`	
	FROM	Mandatory. String. Default: GMT
	TO	Optional. String.
CREATION_USER	Name of the user that created the job. Optional. `CREATION_USER FROM="emuser" TO="em1"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_DATE	Date that the job was created. Optional. `CREATION_DATE FROM="1212" TO="2012"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_TIME	Time that the job was created. Optional. `CREATION_TIME FROM="1230" TO="1430"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CHANGE_USERID	Name of the user that last modified the job. Optional. `CHANGE_USERID FROM="emuser" TO="emacct"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CHANGE_DATE	Date that the job was last modified. Optional. `CHANGE_DATE FROM="1204" TO="1304"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CHANGE_TIME	Time that the job was last modified. Optional. `CHANGE_TIME FROM="1650" TO="1700"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Arguments file parameters for SMART folders

The following table lists the arguments file parameters for SMART folders:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td colspan="3">The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the .dtd file.</td></tr>
<tr><td>*UPDATE*</td><td colspan="2">These tags indicate the start and end of the UPDATE argument. Only criteria located between the tags are considered to be part of the argument. Mandatory.</td></tr>
<tr><td>*SMART_FOLDER*</td><td colspan="2">These tags indicate the start and end of the SMART Folder specification. Criteria identifying the SMART Folders to be modified and indicating the types of modifications to be made are located between these tags. Optional.</td></tr>
<tr><td rowspan="2">*FOLDER_NAME*</td><td colspan="2">Name of the SMART Folder.
- This parameter cannot be modified.
- At least one of the following folder parameters must be included in the input file: DATACENTER, FOLDER_NAME, FOLDER_DSN
`FOLDER_NAME FROM="Tbl42"`</td></tr>
<tr><td>FROM</td><td>String. Mandatory</td></tr>
<tr><td rowspan="2">*DATACENTER*</td><td colspan="2">Name of the Control-M installation to which the job belongs. Optional.
- A TO subparameter cannot be specified for this parameter.
- At least one of the following folder parameters must be included in the input file: DATACENTER, FOLDER_NAME, FOLDER_DSN
`DATACENTER FROM="CTMNYC"`</td></tr>
<tr><td>FROM</td><td>String. Mandatory</td></tr>
<tr><td rowspan="2">*FOLDER_DSN*</td><td colspan="2">(*z/OS only*) Name of the library that contains the SMART Folder. Optional.
- A TO subparameter cannot be specified for this parameter.
- At least one of the following folder parameters must be included in the input file: DATACENTER, FOLDER_NAME, FOLDER_DSN
`FOLDER_DSN FROM="CTMNYC"`</td></tr>
<tr><td>FROM</td><td>String. Mandatory</td></tr>
</table>

Parameter	Description	
FOLDER_ORDER_ METHOD	Optional. A TO subparameter cannot be specified for this parameter. `FOLDER_ORDER_ METHOD FROM="CTMNYC"`	
	FROM	String. Mandatory
SUB_APPLICATION	Name of the group to which the SMART Folder belongs. Used as a descriptive name for related folders. Optional. `GROUP FROM="Grp_HR" TO="Grp_ACCT"`	
	FROM	String. Mandatory.
	TO	String. Optional.
USED_BY	Optional. `USED_BY FROM="fff" TO="ffg"`	
	FROM	String. Mandatory.
	TO	String. Optional.
USED_BY_CODE	Optional. `USED_BY_CODE FROM="C***" TO="D***"`	
	FROM	String. Mandatory.
	TO	String. Optional.
MODIFIED	Optional. `MODIFIED FROM="1101" TO="1102"`	
	FROM	String. Mandatory.
	TO	String. Optional.
LAST_UPLOAD	Date of the last folder upload. String. Optional. `LAST_UPLOAD FROM="1101" TO="1102"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CHECKSUM	Optional. `CHECKSUM FROM="Y" TO="N"`	
	FROM	String. Mandatory.
	TO	String. Optional.
FOLDER_ID	Name of the folder to which the job belongs. Optional. `FOLDER_ID FROM="Tbl001" TO="Tbl002"`	
	FROM	String. Mandatory.
	TO	String. Optional.
REAL_FOLDERID	Optional. `REAL_FOLDERID FROM="43556" TO="43557"`	
	FROM	String. Mandatory.
	TO	String. Optional.
JOBNAME	Name of the job processing definition. Optional. `JOBNAME FROM="Job3"`	
	FROM	String. Mandatory.
	TO	String. Optional.
FILE_NAME	Name of the file that contains the job script. Optional. `FILE_NAME FROM="File3" TO="File7"`	
	FROM	String. Mandatory.
	TO	String. Optional.
APPLICATION	Name of the application to which the job’s group belongs. Used as a descriptive name for related jobs. Optional. `APPLICATION FROM="App3""`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
RUN_AS	Owner (user ID) associated with the job. This parameter is used by the Control-M/Server security mechanism. Optional. `OWNER FROM="emuser" TO="emhr"`	
	FROM	String. Mandatory.
	TO	String. Optional.
ADJUST_COND	Indicates whether to ignore prerequisite conditions normally set by predecessor jobs if the relevant predecessor jobs are not scheduled. Optional. Valid values: ▪ 0 (Do not ignore. Default.) ▪ 1 (Ignore relevant prerequisite conditions.) `ADJUST_COND FROM="1" TO="2"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CONFIRM	Indicates whether the job must be manually confirmed by the Control-M/EM user before it runs. Optional. `CONFIRM FROM="0" TO="1"`	
	FROM	Mandatory. Valid values: ▪ 0 (Default) ▪ 1
	TO	Optional. Valid values: ▪ 0 (Default) ▪ 1
PRIORITY	Indicates Control-M job priority. Optional. Two-character alphanumeric from 00 to ZZ. `PRIORITY FROM="AA" TO="BB"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
TIMEFROM	Indicates the earliest time for submitting the SMART Folder. Format: hhmm. Optional. `TIMEFROM FROM="1430" TO="1450"`	
	FROM	String. Mandatory.
	TO	String. Optional.
TIMETO	Indicates the latest time for submitting the SMART Folder. Format: hhmm. Optional. `TIMETO FROM="1430" TO="1450"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DUE_OUT	Time that the job is expected to finish. Optional. `DUE_OUT FROM="1500" TO="1750"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DOCMEM	Name of the file containing job documentation. String. Optional. `DOCMEM FROM="mem4" TO="Mem67"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DOCLIB	Name of a library or directory containing the job documentation file. String. Optional. `DOCLIB FROM="AcctFiles" TO="HRFiles"`	
	FROM	String. Mandatory.
	TO	String. Optional.
DESCRIPTION	Free text description of the job. String. Optional. `DESCRIPTION FROM="backup jobs from 120399" TO="backup jobs from 021400"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CREATED_BY	Control-M/EM user who defined the job. String. Optional. Example: *CREATED_BY FROM="emuser" TO="emadmin"* The New Day Procedure compares the Author and Owner for each job to check if the job's user has authorization to submit the job. Control-M/EM security levels determine who can edit the Author value (any user or administrators only). For more information, see the Security chapter and the description of the **AuthorSecurity** system parameter in GUI Server parameters.	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_USER	Name of the user that created the job. String. Optional. `CREATION_USER FROM="emuser" TO="em1"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_DATE	Date that the SMART Folder was created. String. Format: ddmm. Optional. `CREATION_DATE FROM="1212" TO="2012"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CREATION_TIME	Time the SMART Folder was created. String. Format: hhmm. Optional. `CREATION_TIME FROM="1230" TO="1430"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CHANGE_USERID	Name of the user that last modified the SMART Folder. String. Optional. `CHANGE_USERID FROM="emuser" TO="emadmin"`	
	FROM	String. Mandatory.
	TO	String. Optional.

Parameter	Description	
CHANGE_DATE	Date that the SMART Folder was last modified. String. Format: ddmm. Optional. `CHANGE_DATE FROM="1204" TO="1304"`	
	FROM	String. Mandatory.
	TO	String. Optional.
CHANGE_TIME	Time that the SMART Folder was last modified. String. Format: hhmm. Optional. `CHANGE_TIME FROM="1650" TO="1700"`	
	FROM	String. Mandatory.
	TO	String. Optional.

updatedef arguments file example

The following example argument files are used with the updatedef utility:

Modify SMART_FOLDER parameter

In the TEST data center, the SMART Folder name FOLDER UNIXJobs is changed to FOLDER TandemJobs.

```
<UPDATE>
<SMART_FOLDER>
<DATACENTER FROM="TEST"/>
<FOLDER FROM="UNIXJobs" TO="TandemJobs"/>
</SMART_FOLDER>
</UPDATE>
```

Modify Folder Name parameter

In the TEST data center, for jobs with FOLDER_ID 12202, the folder name is changed from Tbl_1 to Tbl_2.

```
<UPDATE>
<FOLDER>
<DATACENTER FROM="TEST"/>
<FOLDER_ID FROM="12202"/>
<FOLDER_NAME FROM="Tbl_1" TO="Tbl_2"/>
</FOLDER>
</UPDATE>
```

Modify the Job name of a job

```
<UPDATE>
<JOB>
<FOLDER_NAME FROM="SGMPM1"/>
<!-- <FOLDER_DSN FROM=""/> -->
<DATACENTER FROM="snow"/>
<JOBNAME FROM="cnn*" TO="bbc*"/>
<!-- <FILE_NAME FROM="Job2"/>-->
<!-- <SUBAPPLICATION FROM=""/> -->
<!-- <APPLICATION FROM=""/> -->
<!-- <TASKTYPE FROM=""/> -->
<!-- <AUTHOR FROM=""/> -->
<!-- <MEMLIB FROM=""/> -->
<!-- <CMDLINE FROM="*end" TO="THE END *"/>-->
<!-- <HOSTID FROM=""/> -->
<!-- <RUN_AS FROM=""/> -->
<!-- <MAXRERUN FROM=""/> -->
<!-- <TIMEFROM FROM=""/> -->
<!-- <TIMETO FROM=""/> -->
<!-- <DUE_OUT FROM=""/> -->
<!-- <PRIORITY FROM=""/> -->
<!-- <CRITICAL FROM=""/> -->
<!-- <CYCLIC FROM=""/> -->
<!-- <CONFIRM FROM=""/> -->
<!-- <AUTOARCH FROM=""/> -->
<!-- <INTERVAL FROM=""/> -->
<!-- <OVERRIDE_PATH FROM=""/> -->
<!-- <MAXWAIT FROM=""/> -->
<!-- <DESCRIPTION FROM=""/> -->
<!-- <DOCMEM FROM="docmem"/> -->
<!-- <DOCLIB FROM="doclib"/>-->
<!-- <DAYS FROM=""/> -->
<!-- <DAYS_AND_OR FROM=""/> -->
<!-- <WEEKDAYS FROM=""/> -->
<!-- <DATE FROM=""/> -->
```

```
<!-- <DAYSCAL FROM=""/> -->
<!-- <WEEKSCAL FROM=""/> -->
<!-- <CONFCAL FROM=""/> -->
<!-- <RETRO FROM=""/> -->
<!-- <SHIFT FROM=""/> -->
<!-- <SHIFTNUM FROM=""/> -->
<!-- <MAXDAYS FROM=""/> -->
<!-- <MAXRUNS FROM=""/> -->
<!-- <RERUNMEM FROM=""/> -->
<!-- <RETEN_DAYS FROM=""/> -->
<!-- <RETEN_GEN FROM=""/> -->
<!-- <TASK_CLASS FROM=""/> -->
<!-- <PREV_DAY FROM=""/> -->
<!-- <IND_CYCLIC FROM=""/> -->
<!-- <RULE_BASED_CALENDAR_RELATIONSHIP FROM=""/> -->
<!-- <SYSDB FROM=""/> -->
<!-- <PDSNAME FROM=""/> -->
<!-- <MINIMUM FROM=""/> -->
<!-- <CATEGORY FROM=""/> -->
<!-- <PREVENTNCT2 FROM=""/> -->
<!-- <JAN FROM=""/> -->
<!-- <FEB FROM=""/> -->
<!-- <MAR FROM=""/> -->
<!-- <APR FROM=""/> -->
<!-- <MAY FROM=""/> -->
<!-- <JUN FROM=""/> -->
<!-- <JUL FROM=""/> -->
<!-- <AUG FROM=""/> -->
<!-- <SEP FROM=""/> -->
<!-- <OCT FROM=""/> -->
<!-- <NOV FROM=""/> -->
<!-- <DEC FROM=""/> -->
<!-- <OPTION FROM=""/> -->
<!-- <PAR FROM=""/> -->
<!-- <FROM FROM=""/> -->
```

```
<!-- <ADJUST_COND FROM=""/> -->
<!-- <JOBS_IN_GROUP FROM=""/> -->
<!-- <LARGE_SIZE FROM=""/> -->
<!-- <CREATION_USER FROM=""/> -->
<!-- <CREATION_DATE FROM=""/> -->
<!-- <CREATION_TIME FROM=""/> -->
<!-- <CHANGE_USERID FROM=""/> -->
<!-- <CHANGE_DATE FROM=""/> -->
<!-- <CHANGE_TIME FROM=""/> -->
<!-- <JOB_RELEASE FROM=""/> -->
<!-- <JOB_VERSION FROM=""/> -->
<!-- <FOLDER_ORDER_ METHOD FROM=""/> -->
<!-- <SCHEDULING_ENVIRONMENT FROM=""/> -->
<!-- <SYSTEM_AFFINITY FROM=""/> -->
<!-- <REQUEST_NJE_HOST FROM=""/> -->
<!-- <APPL_TYPE FROM=""/> -->
<!-- <APPL_VER FROM=""/> -->
<!-- <APPL_FORM FROM=""/> -->
<!-- <CM_VER FROM=""/> -->
<!-- <MULTY_AGENT FROM=""/> -->
<!-- <ACTIVE_FROM FROM=""/> -->
<!-- <ACTIVE_TILL FROM=""/> -->
<!-- <TIMEZONE FROM=""/> -->
</JOB>
</UPDATE>
```

emdef utility for services

The emdef is a command line utility used to make various modifications to service definitions in the Control-M/EM database. The emdef uses the following parameters:

Utility Type	Description
defservice (on page 156)	imports service processing definitions into the Control-M/EM database
exportdefservice (on page 162)	exports service definitions

For emdef parameters related to jobs, see emdef utility for jobs (on page 40).

For emdef parameters related to folders and calendars see emdef utility for folders and calendars (on page 245).

defservice

The defservice utility imports service processing definitions into the Control-M/EM database. To run the defservice utility, see Running the defservice utility (on page 156).

defservice reads service processing definitions from a plain text input file written in XML format.

The defservice input file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors. For more information, see defservice input file example (on page 158).

Running the defservice utility

This procedure describes how to run the defservice utility, which enables you to import service processing definitions into the Control-M/EM database.

➢ To run the defservice utility:

1. Do one of the following:
   a. Log on to a Control-M for Databases account (*UNIX*)
   a. -Open a command prompt window (*Windows*) where Control-M/EM is installed.
   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter either of the following commands:

   -
     ```
     emdef defservice [-u <user name> [-p <password>] | -pf <password file>]
     -s <GUI Server Name> -src <xml File Name> [/a] [/o]
     ```

   -
     ```
     emdef defservice [-USERNAME <user name> [-PASSWORD <password>] |
     -PASSWORD_FILE <password file>] -HOST <GUI Server Name> -SRC_FILE <xml
     File Name> [/a] [/o]
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the defservice parameters and switches, see defservice parameters (on page 157) and defservice switches (on page 157).

defservice parameters

The following table describes the defservice utility parameters:

Parameter	Description
<*user name*>	Control-M/EM user name.
<*password*>	Control-M/EM user password.

<*password file*>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName*>
	password=<*password*>

<*GUI Server Name*>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<*XML file name*>	The path and name of the XML file containing the defservice specifications. For more information, see XML file preparation (on page 616)

defservice switches

he following table describes optional switches for the defservice utility:

Switch	Description
?	Displays utility's description and available options.
/a	Accept all. The /a switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) service definitions whose Author does not match the currently logged in user. The /a switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to 2 or 3.
/v	Used to receive verbose messages.

defservice input file example

The following example input file is used with the defservice utility:

```
T<?xml version='1.0' encoding='ISO-8859-1' ?>
<!DOCTYPE DEFSERVICE SYSTEM "defservice.dtd">
<DEFSERVICE >
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100819224237UTC"
INSTANTIATION_TYPE="Filter" LAST_UPDATE_TIME="20100822055018UTC"
NAME="example" ORDERABLE="0" SERVICE_ID="8">
<FILTER >
<INCLUDING_TERMS >
<TERM >
<PARAM NAME="CONTROL-M Name" OP="NEQ" VALUE="isvm-w23-ctmDS9"/>
 </TERM>
</INCLUDING_TERMS>
</FILTER>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100818093723UTC"
INSTANTIATION_TYPE="Filter" LAST_UPDATE_TIME="20100822055952UTC"
NAME="filter service" ORDERABLE="0" SERVICE_ID="1">
<FILTER >
<INCLUDING_TERMS >
<TERM >
<PARAM NAME="CONTROL-M Name" OP="LIKE" VALUE="isvm-w23-ctmDS9"/>
<PARAM NAME="ORDER_FOLDER" OP="LIKE" VALUE="TestCon, TestCon2"/>
</TERM>
</INCLUDING_TERMS>
 </FILTER>
 </SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100818093756UTC"
INSTANTIATION_TYPE="FilterODAT" LAST_UPDATE_TIME="20100822055018UTC"
NAME="odat service" ORDERABLE="0" SERVICE_ID="2">
<FILTER >
<INCLUDING_TERMS >
 <TERM >
<PARAM NAME="CONTROL-M Name" OP="LIKE" VALUE="isvm-w23-ctmDS9"/>
 <PARAM NAME="ORDER_FOLDER" OP="LIKE" VALUE="TestCon, TestCon2"/>
```

```
</TERM>
 </INCLUDING_TERMS>
</FILTER>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100818094142UTC"
INSTANTIATION_TYPE="Job" LAST_UPDATE_TIME="20100822055018UTC" NAME="order
(job) service" ORDERABLE="1" SERVICE_ID="4">
 <FILTER >
<INCLUDING_TERMS >
<TERM >
 <PARAM NAME="CONTROL-M Name" OP="EQ" VALUE="isvm-w23-ctmDS9"/>
<PARAM NAME="ORDER_FOLDER" OP="EQ" VALUE="TestCon_"/>
<PARAM NAME="JOBNAME" OP="EQ" />
</TERM>
</INCLUDING_TERMS>
 </FILTER>
<ORDERABLE_PARAM DISPLAY_NAME="p1" PARAM_NAME="p1" PARAM_TYPE="String"
REQUIRED="1" VALIDATION=","/>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100819211228UTC"
INSTANTIATION_TYPE="SMARTFolder" LAST_UPDATE_TIME="20100822055018UTC"
NAME="tzahi" ORDERABLE="1" SERVICE_ID="6">
<FILTER >
<INCLUDING_TERMS >
<TERM >
<PARAM NAME="CONTROL-M Name" OP="EQ" VALUE="isvm-w23-ctmDS9"/>
<PARAM NAME="ORDER_FOLDER" OP="EQ" VALUE="aaa22"/>
</TERM>
</INCLUDING_TERMS>
</FILTER>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100822045059UTC"
INSTANTIATION_TYPE="Filter" LAST_UPDATE_TIME="20100822055018UTC"
NAME="tzahi10" ORDERABLE="0" SERVICE_ID="10">
<FILTER >
<INCLUDING_TERMS >
<TERM >
```

```
<PARAM NAME="CONTROL-M Name" OP="LIKE" VALUE="isvm-w23-ctmDS9"/>
<PARAM NAME="APPLICATION" OP="LIKE" VALUE="DailyAppl"/>
<PARAM NAME="GROUP" OP="LIKE" VALUE="DailyJobs"/>
<PARAM NAME="ORDER_FOLDER" OP="LIKE" VALUE="Con2Folder"/>
<PARAM NAME="ORDER_LIB" OP="LIKE" VALUE="folder"/>
<PARAM NAME="MEMLIB" OP="LIKE" VALUE="memlib"/>
<PARAM NAME="MEMNAME" OP="LIKE" VALUE="memname"/>
<PARAM NAME="JOBNAME" OP="LIKE" VALUE="jobname"/>
<PARAM NAME="TASKTYPE" OP="LIKE" VALUE="*Job*, *Task*, Command, Detached,
Dummy, External"/>
<PARAM NAME="OWNER" OP="LIKE" VALUE="owner"/>
<PARAM NAME="DEF_HOSTGROUP" OP="LIKE" VALUE="hostgroup"/>
<PARAM NAME="APPL_TYPE" OP="LIKE" VALUE="OS"/>
</TERM>
</INCLUDING_TERMS>
</FILTER>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100819223946UTC"
INSTANTIATION_TYPE="Job" LAST_UPDATE_TIME="20100822055018UTC" NAME="tzahi2"
ORDERABLE="1" SERVICE_ID="7">
<FILTER >
<INCLUDING_TERMS >
<TERM >
<PARAM NAME="CONTROL-M Name" OP="EQ" VALUE="isvm-w23-ctmDS9"/>
<PARAM NAME="ORDER_FOLDER" OP="EQ" VALUE="DailySchedule"/>
<PARAM NAME="JOBNAME" OP="EQ" VALUE="SuadCommand0"/>
</TERM>
</INCLUDING_TERMS>
</FILTER>
<ORDERABLE_PARAM DISPLAY_NAME="param" PARAM_NAME="My Param"
PARAM_TYPE="String" REQUIRED="0" VALIDATION=","/>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100821140755UTC"
INSTANTIATION_TYPE="Filter" LAST_UPDATE_TIME="20100822055018UTC"
NAME="tzahi3" ORDERABLE="0" SERVICE_ID="9">
<FILTER >
 <INCLUDING_TERMS >
```

```
<TERM >
<PARAM NAME="CONTROL-M Name" OP="LIKE" VALUE="isvm-w23-ctmDS9, THID0106"/>
<PARAM NAME="APPLICATION" OP="LIKE" VALUE="SuadDailyAppl*"/>
<PARAM NAME="GROUP" OP="LIKE" VALUE="DailyJobs, TestCon*"/>
<PARAM NAME="MEMLIB" OP="LIKE" VALUE="memlib"/>
<PARAM NAME="TASKTYPE" OP="LIKE" VALUE="*Job*, Command"/>
<PARAM NAME="DEF_HOSTGROUP" OP="LIKE" VALUE="hostgroup"/>
</TERM>
</INCLUDING_TERMS>
</FILTER>
</SERVICE>
<SERVICE CREATED_BY="emuser" CREATE_TIME="20100822060917UTC"
INSTANTIATION_TYPE="Filter" LAST_UPDATE_TIME="20100822060917UTC"
NAME="ServiceFilter" ORDERABLE="0" SERVICE_ID="11">
<FILTER >
<INCLUDING_TERMS >
<TERM >
<PARAM NAME="CONTROL-M Name" OP="LIKE" VALUE="controlm"/>
<PARAM NAME="APPLICATION" OP="LIKE" VALUE="application"/>
<PARAM NAME="GROUP" OP="LIKE" VALUE="group"/>
<PARAM NAME="ORDER_FOLDER" OP="LIKE" VALUE="folder"/>
 <PARAM NAME="ORDER_LIB" OP="LIKE" VALUE="folderlin"/>
<PARAM NAME="MEMLIB" OP="LIKE" VALUE="memlib"/>
<PARAM NAME="MEMNAME" OP="LIKE" VALUE="memname"/>
<PARAM NAME="JOBNAME" OP="LIKE" VALUE="jobname"/>
<PARAM NAME="TASKTYPE" OP="LIKE" VALUE="*Job*, *Task*, Command, Detached,
Dummy, External"/>
<PARAM NAME="OWNER" OP="LIKE" VALUE="owner"/>
<PARAM NAME="DEF_HOSTGROUP" OP="LIKE" VALUE="hostgroup"/>
<PARAM NAME="APPL_TYPE" OP="LIKE" VALUE="OS"/>
<PARAM NAME="DESCRIPTION" OP="LIKE" VALUE="description"/>
</TERM>
</INCLUDING_TERMS>
</FILTER>
</SERVICE>
</DEFSERVICE>
```

exportdefservice

The exportdefservice parameter of the emdef utility exports service processing definitions from the Control-M/EM database to an output file. To run the exportdefservice utility, see Running the exportdefservice utility (on page 162).

When exportdefservice is invoked, it processes a specified file of arguments in XML format. For more information, see exportdefservice arguments file (on page 164). This file contains statements that identify existing service processing definitions. The identified definitions are exported from the Control-M/EM database to an output file. You can modify the exported service processing definitions in the output file and can import the modified definitions into the Control-M/EM database using the defservice (on page 156) utility.

Running the exportdefservice utility

This procedure describes how to run the exportdefservice utility, which enables you to export service processing definitions from the Control-M/EM database to an output file

➢ To run the exportdefservice utility:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter one of the following commands:

-
  ```
  emdef exportdefservice [-u <user> [-p <password>] | -pf <password File>]
  -s <GUI Server Name> -arg <args file name> -out <file name>
  ```

-
  ```
  emdef exportdefservice [-USERNAME <user> [-PASSWORD <password>] |
  -PASSWORD_FILE <password file>] -HOST <GUI Serve rName> -ARG_FILE <args
  file name> -OUT_FILE <file name>
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the exportdefservice parameters and switches, see exportdefservice parameters (on page 163) and exportdefservice switches (on page 164).

exportdefservice parameters

The following table lists the exportdefservice parameters:

Parameter	**Description**
<user name>	Control-M for Databases user name.
<password>	The Control-M for Databases user password.

<password file name>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

<GUI Server Name>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<Args file name>	Path and name of the arguments file containing exportdefservice specifications. For information about this file, see XML file preparation (on page 616)
<file name>	Path and name of the file containing the exported service specifications.
-ctm	Name of the Control-M installation that processes the job.
-folder	Name of folder.
-app	Name of the application to which the job's group belongs
-subapp	Name of the group to which the job belongs.
-service	Name of the service for export.

exportdefservice switches

The following table describes optional switches for the exportdefservice utility:

Switch	Description
/?	Displays utility's description and available options.
/a	Accept all. The **/a** switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The **/a** switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to **2** or **3**.

exportdefservice arguments file

The following rules apply to the exportdefservice argument file:

- More than one service can be specified in an exportdefservice file.
- The arguments file is case-sensitive.
- All parameter values must be enclosed in quotation marks.
- More than one PARAM parameter can be used in a TERM statement.
- The relationship between PARAM parameters in a TERM statement is AND.

The exportdefservice arguments file is checked and processed. If there are any errors, a message is displayed specifying the lines with the errors. The exported service processing definitions are saved to the output file whose name and location is specified in the **-out** <*out file name*> parameter.

For more information on the input file parameters for the exportdefservice utility, see exportdefservice arguments file parameters (on page 165), and exportdefservice arguments file example (on page 166).

exportdefservice arguments file parameters

The following table describes the exportdefservice arguments file parameters:

Parameter	Description	
The first two lines of the arguments file specify the XML version, the text encoding format being used, and the location of the .dtd file.		
TERMS	These tags indicate the start and end of the TERMS file. Only criteria that are located between the tags are considered to be part of the argument.	
TERM	The TERM tags indicate the start and the end of a group of selection criteria used to specify a service or services that are to be exported. Only PARAM tags that are located between the TERM tags are considered to be part of the TERM argument.	
	REL	Relationship between terms. Optional. Valid values: ▪ AND ▪ OR
PARAM	Selection criteria parameter used to determine the service definitions that are to be exported. More than one PARAM can be specified. Mandatory. `PARAM NAME="DATACENTER" OP="E" VALUE="Center1"`	
	NAME	String. Mandatory. The parameter name of any service processing definition parameter.
	OP	Relationship between the NAME and VALUE parameters of the TERM. Mandatory. Valid values: ▪ EQ – equal ▪ NEQ – not equal ▪ LIKE – mask or pattern
	VALUE	String. Mandatory. Value of the parameter specified in the NAME field.

exportdefservice arguments file example

The following example arguments file is used with the exportdefservice utility:

```
<TERMS>
      <TERM>
            <PARAM NAME="NAME" OP="LIKE" VALUE="*"/>
      </TERM>
</TERMS>
```

Control-M/Server utilities

This table lists the Control-M/Server utilities that are used for definition, ordering, and monitoring.

Utility Type	Description
ctmcreate (on page 168)	The ctmcreate utility is an API (Application Program Interface) that allows a specific purpose job to be inserted directly into the Active Jobs database.
ctmdefine (on page 179)	The ctmdefine utility is an API (Application Program Interface) that adds a job processing definition
ctmexdef (on page 189)	The ctmexdef utility exports job processing definitions from the Control-M/Server database to a flat (ASCII) file.
ctmfw File Watcher utility (on page 191)	The ctmfw (Control-M File Watcher) utility monitors file status and detects the following file processes: ▪ Successful completion of a file transfer activity ▪ Creation of a file ▪ Deletion of a file
ctmimptb (on page 203)	The ctmimptb utility imports job definition folders (including SMART Folders and Rule-Based Calendars) that were exported from Control-M/EM by the exportdeffolder utility.
ctmkilljob (on page 206)	The ctmkilljob utility terminates a specified Control-M job and all its processes. ctmkilljob terminates only jobs that are currently executing.
ctmorder (on page 208)	The ctmorder utility orders or forces one or more jobs from a SMART Folder in the Control-M/Server database.
ctmpsm (on page 215)	The ctmpsm utility can be invoked interactively to display the Control-M Production Support menu.
ctmsweep (on page 236)	The ctmsweep utility deletes job definitions from the Control-M/Server database that become obsolete due to the Start date (Active From Date) and End date (Active To Date) parameters in the job definitions.
ctmwhy (on page 240)	The ctmwhy utility displays a report stating why a SMART Folder, Sub-folder, or job waiting in the Active Jobs database is not being submitted for execution.

ctmcreate

The ctmcreate utility is an API (Application Program Interface) that allows a specific purpose job to be inserted directly into the Active Jobs database. The job does not have to be defined in the Control-M/Server database. The function performed by this utility is equivalent to the Force function in Control-M/EM. To run the ctmcreate utility, see Running the ctmcreate utility (on page 168).

The ctmcreate utility can be invoked using the -input_file parameter:

```
ctmcreate -input_file <fullPathFileName>
```

The referenced file contains all the required parameters. Most of the parameters are described in Control-M Parameters and ctmcreate parameters *(on page* 171*).*

For more information on syntax rules, see ctmcreate syntax rules (on page 175).

You can create a job within the given folder or Sub-folder name. The ctmcreate utility can be used to create a specific purpose (non-permanent) folder that is not defined in the Control-M/Server database by specifying:

```
-what FOLDER|SUBFOLDER -folder <folderName|folderPath>
```

Running the ctmcreate utility

This procedure describes how to run the ctmcreate utility, which enables you to insert a specific purpose job directly into the Active Jobs database.

➢ To run the ctmcreate utility:

- Type the following command:

```
ctmcreate
  -TASKTYPE
<JOB|EXTERNAL|DETACHED|COMMAND|DUMMY|FOLDER|SUBFOLDER>
   [ -SUB_APPLICATION           <sub_application name> ]
   [ -APPLICATION     <applic name> ]
   [ -DEBUG           <debug level 0-5> ]
   [ -QUIET  ]
   [ -FOLDER_ORD       <Active or Sub folder orderno|ALONE|LAST> ]
   [ -ADJUST_COND     Y|N ]
   [ -MULTIAGENT      Y|N ]
   [ -HOSTGRP         <name> ]
   [ -MEMLIB          <path> ]
   [ -MEMNAME         <filename> ]
   [ -CMDLINE         <string> ]
   [ -EMBEDDED_SCRIPT <file name> ]
   [ -JOBNAME         <name> ]
   [ -FOLDER           <name> ]
```

```
     [ -RUN_AS           <username> ]
     [ -CREATED_BY          <username> ]
     [ -ODATE            <date>|ODAT ]
     [ -ODATE_OPTION     VALUE_DATE|RUN_DATE ]
     [ -MAXRERUN         <value> ]
     [ -TIMEZONE         <xxx> ]
     [ -TIMEFROM         <earliest submission time> ]
     [ -TIMEUNTIL        <latest  submission time> | '>' ]
     [ -PRIORITY         <job priority> ]
     [ -CRITICAL         Y|N ]
     [ -CYCLIC           Y|N ]
     [ -CYCLIC_TYPE      INTERVAL|INTERVAL_SEQUENCE|SPECIFIC_TIMES ]
     [ -SPECIFIC_TIMES  <specific times string (HHMM,HHMM)> ]
     [ -INTERVAL_SEQUENCE  <interval sequence string e.g(+1H,+2M)> ]
     [ -TOLERANCE    <maximum delay allowed (minutes)> ]
     [ -CONFIRM          Y|N ]
     [ -APPLTYPE         <agent_application> ]
     [ -APPLVER          <application version> ]
     [ -CMVER            <CM version> ]
     [ -APPLFORM         <application form> ]
     [ -INTERVAL         <45d(days) | 1080h(hours) | 64800m (minutes)> ]
     [ -INTERVALFROM     START | END | TARGET ]
     [ -OVERRIDE_PATH          <alternative directory> ]
     [ -MAXWAIT          <days> ]
     [ -DESCRIPTION      <string> ]
     [ -DOCMEM           <filename> ]
     [ -DOCLIB           <directory name> ]
     [ -INCOND           <condition> <dateref>|ODAT  AND|OR ]
     [ -OUTCOND          <condition> <dateref>|ODAT  ADD|DEL ]
     [ -VARIABLE         <varname> <expression> ]
     [ -QUANTITATIVE     <name> <quantity> ]
     [ -OUTPUT           RELEASE|DELETE|COPY|MOVE [<parameter>]]
     [ -Control          <name>  E|S ]
     [ -SHOUT            OK|NOTOK|RERUN|LATESUB|LATETIME|EXECTIME
   <destination> <urgency R|U|V> <message> [<time>] ]
```

```
     [ -ON              <statement> <code>
         [ -DOOK ]
         [ -DONOTOK ]
         [ -DORERUN ]
         [ -DOSHOUT    <destination> <urgency R|U|V> <message> ]
         [ -DOCOND     <condname> <dateref>|ODAT  ADD|DEL ]
         [ -DOVARIABLE <varname> <expression> ]
         [ -DOFORCEJOB <foldername> <jobname> <odate>|ODAT ]
         [ -DOOUTPUT    RELEASE|DELETE|COPY|MOVE [<parameter>] ] ]
         [ -DOSTOPCYCLIC ]
         [ -DOMAIL <destination> <cc> <urgency R|U|V> <subject> <message>
   [<attach output>] ]
         [ -DOREMEDY <summary> <description> <urgency L|M|H|U|C>
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmcreate parameters, see ctmcreate parameters (on page 171).

ctmcreate parameters

The following table describes ctmcreate utility parameters.All other parameters (for example, cyclic job parameters) are described in Control-M Parameters.

Parameter	Description
-APPLICATION	Provides a logical name for sorting groups of jobs. This parameter is used to supply a common descriptive name to a set of related groups of jobs.
-CREATED_BY	Control-M/EM user who defined the job. String up to 64 characters. Optional. This argument is used by the Control-M/Server security mechanism and, under certain circumstances, cannot be modified. For more information, see the Security chapter and the description of the AuthorSecurity system parameter in GUI Server parameters.
-CYCLIC	Indicates whether the job is cyclic (to be run at regular intervals). Optional. Valid values: ▪ Y – Yes ▪ N – No (Default)
-CYCLIC_TYPE	Determines the type of cyclic job: ▪ Interval ▪ Interval Sequence ▪ Specific Times
-INTERVAL SEQUENCE	A list of time intervals (for example +30M,+2H,+1D) up to 4000 characters including commas. Value range: ▪ Minutes: 0-64,800 ▪ Hours: 0-1080 ▪ Days: 0-45
-SPECIFIC_TIMES	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730).
-ODATE	Indicates the scheduling date (odate) to be associated with job(s). Valid values are:

Parameter	Description	
	ODAT	The current working date of the computer on which Control-M/Server is running. This is the default value.
	yyyymmdd	A specific working day in *yyyymmdd* format.
	The interpretation of this parameter value is dependent on the value specified for the -odate_option parameter (described below).	
-ODATE_OPTION	Indicates how the specified -odate value should be used. Valid values are:	
	value_date	The specified odate is the odate value for the job. However, the job should be run during the current working day. This is the default value for the -odate_option parameter. If a time zone is specified in the job processing definition, then the job is run according to those time zones.
	run_date	The jobs that are ordered by this run of the ctmcreate utility should be run only when the specified odate begins. If the specified odate is the current working day, this job will work in the same way as value_date (described above). If the specified odate has not begun (for example, due to time zone specifications), then the job will wait in the Active Jobs database (with WAIT_ODAT status) until the start of the specified working day. If the specified odate has already passed, the ctmcreate utility will not run, and an error message will be displayed.
-FOLDER_ORD	Specifies in which order of a SMART Folder to put the job. Valid values are:	
	<order-no>	A specific order number of the SMART Folder. If the specified order number does not exist the command is not executed and an error message is displayed.

Parameter	Description	
	ALONE	The job is on its own (not in any folder).
	LAST	The last order of the specified SMART Folder.
	When an order ID or LAST is specified for this parameter, the -folder parameter is mandatory and must contain the name of a SMART Folder that is currently in the Active Jobs database. If more than one folder already exists while creating a sub-folder in the Active Jobs database and the -FOLDER_ORD option is not specified, the folder with highest order number is chosen.	
-EMBEDDED_SCRIPT	The Embedded script parameter contains the name of the file and path, which enables the embedded script to be copied from a file.	
-DEBUG	Level of debug messages, 0 to 5. Default: 0 (no debug messages).	
-QUIET	Indicates, if specified, that no informational messages are displayed during the execution of the command.	
-INPUT_FILE	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: prepare and save files of utility parameters that can be reused. specify utility input longer than the number of characters allowed in the command line. `-input_file ~<controlm_owner>/ctm_server/data/ctm create_parms.txt`	
-DOMAIL	Sends mail when the job run is complete. Optional. `DOMAIL urgency="R" destination="emuser@emuser.com" cc="barry@emuser.com" subject="OK" message="Task completed OK."`	
	destination	Recipient of the message. String. Mandatory.
	cc	Additional recipient of the message. String. Optional.

Parameter	Description	
	urgency	Urgency of the message. Valid values: ▪ R (regular - Default) ▪ U (urgent) ▪ V (very urgent)
	subject	Brief text description of the message contents. String. Optional.
	message	Text of the message. String. Mandatory.
	attach output	Specifies at the job level whether the OUTPUT should be sent as an email attachment. Valid values: ▪ Y – Yes ▪ N – No ▪ D – default (this means take the value from the Control-M/Server configuration file)
-TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Valid range: 0-999	

ctmcreate syntax rules

The following rules apply when using this utility:

- More than one parameter can be specified on a line.
- The odate parameter specifies the date to use as the job's scheduling date. Specify a date in *yyyymmdd* format, or specify ODAT to accept the Control-M system date.
- The %%NEXT, %%$NEXT, %%PREV, and %%$PREV variables cannot be specified for the ctmcreate utility. These variables refer to the next or previous scheduling date and are not relevant for a utility that places jobs directly in the Active Jobs database.
- The length of the command line, after decoding, must not exceed 999 characters.
- Although most parameters are optional, certain parameters are required depending on the value specified for -what.
- On computers that support Disk Clustering, the -hostgrp parameter is required (including either a host group name, or the virtual name of the Control-M/Agent).
- All parameter fields (as specified in the syntax) must contain values. If no value is required, specify a null string "" in the relevant position in the parameter specification.

The -domail parameter has the following syntax:

-domail <*destination*> <*cc*> <*severity*> <*subject*> <*message*>

To specify this command without a value for the cc field, include a null string in the appropriate location. For example:

-domail johnsmith@bmc.com "" R "subject line" "My message"

For more information, see information about setting Control-M/Server e-mail configuration parameters in Email parameters.

- JOB and DETACHED require memname and memlib parameters.
- COMMAND requires the cmdline parameter.
- Strings containing blanks must be enclosed in quotation (for example, -cmdline "ctmudlst list payroll").
- A UNIX metasymbol (that must be enclosed in quotation marks) appearing in a command line string should be enclosed in single quotation (for example, -cmdline "ctmcontb list '*' ").
- If a parameter value begins with a $ sign, the operating system will try to replace the value. For example, -jobname $USER will cause the shell to substitute the current user. If a parameter value should contain a $ sign, enclose the value in single quotation marks. For example, -jobname 'test$' will set the jobname parameter to test$.
- A variable that does not contain a $ sign can be enclosed in single or double quotation marks. A variable that does contain a $ sign should be enclosed in single quotation marks. A variable containing a $ sign cannot be resolved if it is enclosed in double quotation marks.
- Condition dates are specified in *mmdd* format. Time is specified in *hhmm* format.
- A parameter requiring more than one entry can be repeated as many times as necessary (for example, if a job must wait for several prerequisite conditions, specify a separate -incond parameter for each prerequisite condition).

  The following special characters are disabled when they occur in prerequisite condition names:

- (open parenthesis
-) close parenthesis
- | vertical bar
- space

- An -on parameter must be followed by at least one -do... parameter.
- -do... parameters are dependent upon the last -on parameter preceding them.
- Normally, when a -dorerun parameter is implemented, the current run of the job ends with **NOTOK** status. To ensure that the job will have **OK** status even though it is rerun, specify a -dook parameter immediately after the -dorerun parameter.
- The -dorerun parameter cannot be specified for a cyclic job.
- The order of the parameters does not affect the outcome of the job, with the exception of -on and -do... parameters.
- When using -doforcejob to force an entire folder, <job name> must be specified as a blank enclosed in quotation marks (that is, "   ").
- When the ctmcreate utility is invoked from a script, to use the **** option for a -incond date parameter, specify the parameter as \"****\"
- If a single character is specified for the priority parameter, the first character is assumed to be A. For example, priority 1 is interpreted as priority A1.
- A maximum of 99 prerequisite conditions can be specified for docond parameters.

```
-incond pk_oly_ok "****"
```

For more information on ctmcreate parameters , see ctmcreate parameters (on page 171) , and ctmcreate example (on page 176).

ctmcreate example

The following are example usages of the ctmcreate utility:

When the UNIX symbol ~ is used in parameter -memlib, -override path, or -doclib to represent the user's home directory, the entire parameter should be enclosed in double quotation marks. The quotation marks ensure that the ~ symbol will be translated by the agent before submission, and not by the server before transmission to the agent computer.

```
-file_path "~/controlm/scripts/"
```

The following command contains the minimum parameters required to create a job in the Active Jobs database:

```
ctmcreate -what command -group em\
        -application test -cmdline "ls -l /etc/passwd"
```

You can get the same result by using the -input_file parameter as follows:

```
ctmcreate -input_file
~<controlm_owner>/ctm_server/data/ctmcreate_delfr.txt
```

The referenced file contains the following lines:

```
-tasktype command
-subapp em
-application test
-cmdline "ls -l /etc/passwd"
```

The following command includes examples of most of the parameters that can be used to create a job in the Active Jobs database:

```
ctmcreate  -what JOB \
-cyclic N \
 -description "Daily Summary" \
 -subapp SUPPLY   -application SUPPLIES \
  -file_path /users/ctm_server/  -file_name PROLYPAR  -hostgrp 
UNIXGRP \
-jobname PROLYPAR \
 -run_as suppman \
 -odate 19981130 \
 -timeuntil 1800 \
-priority AA -critical N \
-confirm Y \
 -doclib  /users/supply/doc/   -docmem prolypardoc \
-incond pk_oly_ok ODAT AND \
-incond pk_olp_ok ODAT AND \
-outcond pk_oly_ok ODAT DEL \
-outcond pk_olp_ok ODAT DEL \
 -outcond pk_olypar ODAT ADD \
-variable %%PARM1 "%%CALCDATE %%ODATE -2" \
 -quantitative tape 2  -quantitative cpu 50 \
-output MOVE /test/logs/ \
-control disk2 E \
-shout OK oper2 U "Daily summary completed" \
-on "COPY JWINFO_2507" "%COPY-E-OPENIN, error" \
-dooutput MOVE /oper/openerr
```

The following command creates an empty SMART folder called myfolder:

```
ctmcreate -what FOLDER -FOLDER myfolder
```

The response is:

new ORDER created, orderid:00000b(11) for JOBNAME=myfolder.

The following command creates an empty Sub-folder called mysubfolder within the SMART folder called myfolder:

```
ctmcreate -what SUBFOLDER -FOLDER myfolder/mysubfolder
```

The response is:

Attached to the SMART folder 'myfolder': 00000b(11)

new ORDER created, orderid:00000c(12) for JOBNAME=mysubfolder.

ctmdefine

The ctmdefine utility is an API (Application Program Interface) that adds a job processing definition to one of the following in the Control-M/Server database:

- A folder
- A SMART Folder
- A Sub-folder

To run the ctmdefine utility, see Running the ctmdefine utility (on page 180).

This utility can be used when converting job scheduling information from other job control components to Control-M/Server. The function performed by this utility is equivalent to the manual process of creating job processing definitions, described in Job definition.

The ctmdefine utility can be invoked using the -input_file parameter:

```
-ctmdefine -input_file <fullPathFileName>
```

The referenced file contains all the required parameters. Most of the parameters are described in Control-M Parameters and ctmdefine parameters (on page 183).

For more information on ctmdefine syntax rules, see ctmdefine syntax rules (on page 184).

The ctmdefine utility can also be used to define jobs for specific applications, for example, SAP and Oracle Applications (OAP).

When creating new SMART folders or job processing definitions, the following considerations are applicable:

- If the job name specified when using this utility already exists in a job processing definition in the SMART Folder, the new job processing definition does not overwrite the existing one. Both job processing definitions will appear in the folder, each with a different internal job number.
- If the SMART Folder specified when using this utility does not exist, the utility creates it.
- After using this utility to create one or more job processing definitions, download the modified SMART Folders to the Control-M/EM database.
- A newly created SMART folder can be assigned a Order method parameter using the Control-M Workload Automation after the folder is downloaded to the Control-M/EM database or by using the ctmpsm utility.
- Values for a creation/modification timestamp and the user ID of the user who created or modified the job processing definition are automatically added to the communication protocol between Control-M/EM and Control-M/Server and are stored in the Control-M/EM database. These values are initialized by the ctmdefine utility and are sent to Control-M/EM when the SMART Folder is downloaded. When the SMART Folder is uploaded, Control-M/EM sends these values to Control-M/Server.
- When defining a job in a nested folder, a folder path to the intended nested folder must be specified (-FOLDER option). The folder path starts from the outermost folder name and is similar to operating system files and directories path. When a folder name is used and the folder does not exist, a folder is created.
- If you define a new Rule-based calendar with the ! character at the beginning of the Rule-based calendar name, the Rule-based calendar is excluded. If this feature is disabled, an error message is displayed that you cannot define a Rule-based calendar with the ! character. For more information, see **DefaultCTMExcludeRBC** in General parameters.

Running the ctmdefine utility

This procedure describes how to run the ctmdefine utility, which enables you to add a job processing definition to a folder, a SMART folder, and a subfolder in the Control-M/Server database/

➢ To run the ctmdefine utility:

- Type the following command:

```
ctmdefine
  -FOLDER          <name>
  -JOBNAME        <name>
  -WHAT       <JOB|EXTERNAL|DETACHED|COMMAND|DUMMY>
  -SUB_APPLICATION          <sub_application name>
  -APPLICATION    <applic name>
  [ -CMDLINE      <string> ]
  [ -EMBEDDED_SCRIPT <file name> ]
  [ -MAXRERUN     <value> ]
  [ -CRITICAL     Y|N ]
  [ -CYCLIC       Y|N ]
  [ -CYCLIC_TYPE     INTERVAL|INTERVAL_SEQUENCE|SPECIFIC_TIMES ]
  [ -INTERVAL        <45d(days) | 1080h(hours) | 64800m (minutes)> ]
  [ -SPECIFIC_TIMES  <specific times string (HHMM,HHMM)> ]
  [ -INTERVAL_SEQUENCE  <interval sequence string e.g(+1H,+2M)> ]
  [ -TOLERANCE    <maximum delay allowed (minutes)> ]
  [ -INTERVALFROM    START | END | TARGET ]
  [ -OVERRIDE_PATH      <alternative directory> ]
  [ -RELATIONSHIP        AND|OR       ]
  [ -MAXWAIT      <days> ]
  [ -HOSTGRP      <name> ]
  [ -MEMLIB       <path> ]
  [ -MEMNAME      <filename> ]
  [ -MULTIAGENT   Y|N ]
  [ -ADJUST_COND  Y|N ]
  [ -RUN_AS        <username> ]
  [ -CREATED BY       <username> ]
  [ -DEBUG        <debug level 0-5> ]
  [ -QUIET  ]
  [ -TIMEZONE     <xxx> ]
  [ -TIMEFROM     <earliest submission time> ]
  [ -TIMEUNTIL    <latest  submission time> | '>' ]
  [ -PRIORITY     <job priority> ]
  [ -CONFIRM      Y|N ]
  [ -APPLTYPE     <agent_application> ]
  [ -APPLVER      <application version> ]
  [ -CMVER        <CM version> ]
  [ -APPLFORM     <application form> ]
  [ -DESCRIPTION  <string> ]
  [ -DOCMEM       <filename> ]
  [ -DOCLIB       <directory name> ]
  [ -INCOND       <condition> <dateref>|ODAT  AND|OR ]
  [ -OUTCOND      <condition> <dateref>|ODAT  ADD|DEL ]
  [ -VARIABLE     <varname> <expression> ]
  [ -QUANTITATIVE <name> <quantity> ]
  [ -OUTPUT       RELEASE|DELETE|COPY|MOVE [<parameter>]]
  [ -CONTROL      <name>  E|S ]
  [ -SHOUT        OK|NOTOK|RERUN|LATESUB|LATETIME|EXECTIME <destination>
<urgency R|U|V> <message> [<time>] ]
  [ -ON           <statement> <code>
      [ -DOOK ]
      [ -DONOTOK ]
      [ -DORERUN ]
      [ -DOOUTPUT    RELEASE|DELETE|COPY|MOVE [<parameter>] ] ]
      [ -DOSTOPCYCLIC ]
      [ -DOSHOUT    <destination> <urgency R|U|V> <message> ]
      [ -DOCOND     <condname> <dateref>|ODAT  ADD|DEL ]
      [ -DOVARIABLE <varname> <expression> ]
      [ -DOFORCEJOB <foldername> <jobname> <odate>|ODAT ]
      [ -DOMAIL <destination> <cc> <urgency R|U|V> <subject> <message>
[<attach output>] ]
      [ -DOREMEDY <summary> <description> <urgency L|M|H|U|C> ]
[ -DAYS         <daystr> ]
  [ -WEEKDAYS     <weekdaystr> ]
```

```
    [ -MONTH        ALL|JAN|FEB|MAR|APR|MAY|JUN|JUL|AUG|SEP|OCT|NOV|DEC
  Y|N ]
    [ -DATE          <MMDD> ]
    [ -DATEFROM      <YYYYMMDD> ]
    [ -DATEUNTIL     <YYYYMMDD> ]
    [ -DAYSCAL       <calendar> ]
    [ -WEEKCAL       <calendar> ]
    [ -CAL_ANDOR     AND|OR ]
    [ -SHIFT         [</>/@][+/-]nn ]
    [ -CONFCAL       <calendar> ]
    [ -RETRO         Y|N ]
    [ -RBC <rule_based_calendar>    ]
```

If you define a new Rule-based calendar with the ! character at the beginning of the Rule-based calendar name, the Rule-based calendar is excluded. If this feature is disabled, an error message is displayed that you cannot define a Rule-based calendar with the ! character. For more information, see **DefaultCTMExcludeRBC** in General parameters.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmdefine parameters, see ctmdefine parameters (on page 183).

Defining application-specific Jobs

The ctmdefine utility can be used to define jobs for specific applications, for example, SAP and Oracle Applications. These jobs are defined by setting the -appltype parameter to, for example, SAP or OAP.

The -memname and -memlib parameters must also be specified for the ctmdefine utility when defining application-specific jobs.

In addition to these parameters, you can specify application-specific parameters as variables. These variables are described in detail in the SAP and Oracle Applications user guides.

```
ctmdefine -what job -jobname sap1 -MEMNAME test -memlib sap -VARIABLE %%SAPR3-JOB_MODE
          CREATE -VARIABLE %%SAPR3-ACCOUNT DV2 -VARIABLE %%SAPR3-STEP-S01-PROGRAM
          ZQA_SIMPLE -owner sapr3 -VARIABLE %%SAPR3-STEP-S01-STEP_TYPE A -APPLTYPE SAP
          -hostgrp nord -VARIABLE %%SAPR3-JOBNAME sap1 -folder SAP1 -application SAP1 -group
          SAP1
```

ctmdefine parameters

The following table describes debug, quiet, and input_file parameters. All other parameters (for example, cyclic job parameters) are described in Control-M Parameters.

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-APPLICATION</td><td colspan="2">Provides a logical name for sorting groups of jobs. This parameter is used to supply a common descriptive name to a set of related groups of jobs.</td></tr>
<tr><td>-CREATED_BY</td><td colspan="2">Control-M/EM user who defined the job. String up to 64 characters. Optional.
This argument is used by the Control-M/Server security mechanism and, under certain circumstances, cannot be modified. For more information, see the Security chapter and the description of the AuthorSecurity system parameter in GUI Server parameters.</td></tr>
<tr><td>-debug</td><td colspan="2">Level of debug messages, 0 to 5. Default: 0 (no debug messages).</td></tr>
<tr><td>-quiet</td><td colspan="2">If specified, no information messages are displayed during execution of the command.</td></tr>
<tr><td>-input_file</td><td colspan="2">Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to:
prepare and save files of utility parameters that can be reused.
specify utility input longer than the number of characters allowed in the command line.
-input_file ~<*controlmOwner*>/ctm_server/data/ctmdefine_parms.txt</td></tr>
<tr><td rowspan="3">-DOMAIL</td><td colspan="2">Sends mail when the job run is complete. Optional.
DOMAIL urgency="R" destination="emuser@emuser.com"
          cc="barry@emuser.com" subject="OK"
          message="Task completed OK."</td></tr>
<tr><td>destination</td><td>Recipient of the message. String. Mandatory.</td></tr>
<tr><td>cc</td><td>Additional recipient of the message. String. Optional.</td></tr>
</table>

Parameter	Description	
	urgency	Urgency of the message. Valid values: ▪ R (regular - Default) ▪ U (urgent) ▪ V (very urgent)
	subject	Brief text description of the message contents. String. Optional.
	message	Text of the message. String. Mandatory.
	attach output	Specifies at the job level whether the OUTPUT should be sent as an email attachment. Valid values: ▪ Y – Yes ▪ N – No ▪ D – default (this means take the value from the Control-M/Server configuration file)

ctmdefine syntax rules

The following syntax rules apply for this utility:

- More than one parameter can be specified on a line.
- Keywords can be written in either uppercase or lowercase, but parameter values are case sensitive.

-subapp ACCGROUP and -subapp ACCGROUP

specify the same group ACCGROUP.

-subapp accgroup

specifies a different group accgroup.

- For the -month parameter, specify the first three letters of a month (for example, JAN) or ALL for all months (the default is none). To specify two or more individual months, use a separate -month parameter for each month.
- If a single character is specified for the priority parameter, the first character is assumed to be A. For example, priority 1 is interpreted as priority A1.
- The length of the command line, after decoding, must not exceed 999 characters.
- Although most parameters are listed as optional, certain parameters may be required, depending on the option specified for parameter -what.
- All task types require the group and application parameters.
- TASKTYPE JOB and DETACHED require parameters memname and memlib.
- TASKTYPE COMMAND requires parameter cmdline.
- FOLDER requires the RBC (Rule-based calendar) parameters. Each RBC definition is followed by its scheduling parameters. If a Rule-based calendar is defined with the ! character at the beginning of the Rule-based calendar name, the Rule-based calendar is excluded. If this feature is disabled, an error message is displayed that you cannot define a Rule-based calendar with the ! character. For more information, see **DefaultCTMExcludeRBC** in General parameters.

```
-rbc myrbc1
-maxwait 1
-days 0,2,3
-dayscal cal1
```

- Strings containing blanks must be enclosed in quotation marks
  (for example, -cmdline "ctmudlst list payroll").
- A UNIX metasymbol (that must be enclosed in quotation marks) in a command line string should be enclosed in single quotation (for example, -cmdline "ctmcontb list '*' ").
- If a parameter value begins with a $ sign, the operating system will try to replace the value. For example, -jobname $USER will cause the shell to substitute the current user. If a parameter value should contain a $ sign, enclose the value in single quotation marks. For example, -jobname 'test$' will set the jobname parameter to test$.
- A variable that does not contain a $ sign can be enclosed in single or double quotation marks. A variable that does contain a $ sign should be enclosed in single quotation marks. A variable containing a $ sign cannot be resolved if it is enclosed in double quotation marks.
- Condition dates are specified in *mmdd* format. Time is specified in *hhmm* format.
- On computers that support Disk Clustering, the -hostgrp parameter is required (including either a host group name, or the virtual name of the Control-M/Agent).
- A parameter requiring more than one entry can be repeated as needed:
  - If a job is dependent upon several prerequisite conditions, specify a separate -incond parameter for each prerequisite condition.
  - If a job can run only in January and July, specify a separate
    -month parameter for each month. (For example,
    -month ALL N -month JAN Y -month JUL Y.)

  - If a job should run every month except July, specify -month ALL Y and -month JUL N.
  - If a job is in a SMART Folder and is scheduled according to the FIRSTDAY and SALARY1 Rule-Based Calendars, specify –RBC FIRSTDAY –RBC SALARY1.

- The default for the -month parameter is ALL Y, which means that if you want to define a job that should run only in one specific month, you must first indicate that it should not run on any month. For example: -month ALL N -month NOV Y
- An -on parameter must be followed by at least one -do... parameter.
- Additional post-processing conditions can be set by using -on <statement> <code>.

  On statement: *

  Code: RUNCOUNT>4

  Do Shout: **To:** Control-M/EM

  Text: "Job ran more than four times"
- -do... parameters are dependent upon the last -on parameter preceding them.
- The order of parameters does not affect the outcome of the job, with the exception of -on and -do... parameters.
- All fields of each parameter (as specified in the syntax section) must contain values. If no value is required for a parameter field, a null string "" must be specified in the relevant position in the parameter specification.

The -domail parameter has the following syntax:

```
-domail <destination> <cc> <severity> <subject> <message>
```

To specify this command without a value for the cc field, include a null string in the appropriate location. For example:

```
-domail johnsmith@bmc.com "" R "subject line" "My message"
```

- Normally, when a -dorerun parameter is implemented, the current run of the job ends with NOTOK status. To ensure that the job will have an OK status, even though it is rerun, specify a -dook parameter immediately after the -dorerun parameter.
- The -dorerun parameter cannot be specified for a cyclic job.
- When using -doforcejob to force an entire folder, <job name> must be specified as a blank enclosed in quotation marks (that is, "   ").
- IN condition statements that use complex boolean logic can be specified.

  For more information, see the description of the In Condition parameter in *Control-M Workload Automation Parameters*.

- When ctmdefine is invoked from a script: To use the **** option for a -incond date parameter, specify this parameter as "****"

```
-incond pk_oly_ok "****"
```

- When the UNIX symbol ~ is used in a -memlib, -override path, or -doclib parameter to represent the user's home directory, the entire parameter should be enclosed in double quotation marks, which ensures that the ~ will be translated by the agent before submission, and not by Control-M/Server before transmission to the agent computer.

```
-memlib "~/controlm/scripts/"
```

- A maximum of 99 prerequisite conditions can be specified for docond parameters.
- Condition names using both open and closed square brackets ([and]) must be enclosed in quotation marks (for example, "RATE[A1]").

  The following special characters are disabled when they occur in prerequisite condition names:

  - (open parenthesis
  -) close parenthesis
  - | vertical bar
  - space

- The -shift parameter has been extended to four characters (xyyy). The first character (x) indicates how to shift scheduling of the job if the original scheduling day of the job is not a working day in the CONFCAL calendar. Valid values are:

  - "" (Blank) – No shifting occurs. The job is not scheduled. Default.
  - > – Job scheduling is shifted to the next working day in the CONFCAL calendar. Additional shifting may be performed, depending on the yyy value, described below.
  - < – Job scheduling is shifted to the previous working day in the CONFCAL calendar. Additional shifting may or may not be performed, depending on the yyy value, described below.
  - @ – Tentatively schedule the job for the current day, even if the current day is not a working day in the CONFCAL calendar. Additional shifting may or may not be performed, depending on the yyy value, described below.

  The remaining three characters (yyy) shift scheduling of the job forward or backward the specified number of working days, as defined in the CONFCAL calendar. Valid values are:

    - Blank - no shifting occurs
    - -nn or +nn shifts the job forward or backward nn working days in the CONFCAL calendar. nn can be any value from 0 to 62.

  - If the result of shifting by yyy days is a day that is not allowed (-n was entered for that day in the DAYS parameter), the job is shifted to the next working day (for a forward shift), or to the previous working day (for a backward shift).
  - If the original scheduling day of the job is a working day in the CONFCAL calendar, the x value is ignored and the yyy value determines when the job is scheduled.

- If the original scheduling day of the job is not a working day in the CONFCAL calendar, job scheduling is shifted according to the x value and then shifted again according to the yyy value (if specified) to determine when the job is scheduled.
- If the original scheduling day of the job is not a working day in the CONFCAL calendar, and no value (blank) is specified for the x value, the job is not scheduled, and the yyy value (if specified) is ignored.
- Confcal and Shift parameters are applied to a scheduling date only if that date already satisfies the Basic Scheduling criteria as specified in the Days, Months, Dates, and Weekdays parameters.

The following command contains the minimum parameters required to define a job:

```
ctmdefine -folder cmmnds -jobname cmls13 \
-what command  -group em  -application test \
-date 0101  -cmdline "ls -l /etc/passwd"
```

You can get the same result by using the -input_file parameter as follows:

```
ctmdefine -input_file
~<controlm_owner>/ctm_server/data/ctmdefine_cmmnds.txt
```

The referenced file (**ctmdefine_cmmnds.txt**) contains the following lines:

- `-folder cmmnds`
- `-jobname cmls13`
- `-tasktype command`
- `-group em`
- `-application test`
- `-date 0101`
- `-cmdline "ls -l \etc\passwd"`

For more information on the parameters for the ctmdefine utility, see ctmdefine parameters (on page 183).

ctmexdef

The ctmexdef utility exports job processing definitions from the Control-M/Server database to a flat (ASCII) file. This file can then be used as input for either the ctmcreate utility or the ctmdefine utility.

The ctmexdef utility can be used to:

- Modify existing job processing definitions in batch mode (together with the ctmdefine utility). The job processing definitions in the file exported by ctmexdef can be edited offline and then returned to the Control-M/Server database by using ctmdefine. For more information, see ctmdefine (on page 179).
- Creating specific purpose jobs to be inserted in the Active Jobs database based on previously defined jobs. Together with the ctmcreate utility, the ctmexdef utility can copy and modify job processing definitions in batch mode that can then be sent directly to the Active Jobs database by using ctmcreate. See ctmcreate (on page 168) for a complete description of the utility.

To run the ctmexdef utility, see Running the ctmexdef utility (on page 189).

When working on UNIX, the **utf8** file stores the script. No translation is performed by the utility when being read into memory. In Windows, that store the script is native. Translation to utf8 needs to be performed by either the ctmcreate or ctmdefine when being read into memory.

The output of ctmexdef on Windows cannot be used as input on UNIX installations and the other way around.

For more information on ctmexdef syntax rules, see ctmexdef syntax rules (on page 190).

Running the ctmexdef utility

This procedure describes how to run the ctmexdef utility, which enables you to export job processing definitions from the Control-M/Server database to a flat (ASCII) file

➢ To run the ctmexdef utility:

- Type the following command:

```
ctmexdef
  -FOLDER                <name>
  -JOBNAME|-MEMNAME     <name>
 [  -ACTION           <DEFINE|CREATE> ]
 [  -FILE             <filename> ]
 [  -WORKING_DIR      <working directory> ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmexdef parameters, see ctmexdef parameters (on page 190) .

ctmexdef parameters

The following table lists debug, quiet, and input_file parameters for the ctmexdef utility. All other parameters (for example, cyclic job parameters) are described in *Control-M Parameters*.

Parameter	Description
-ACTION	DEFINE – The exported file will be in ctmdefine format. Default.
	CREATE – The exported file will be in ctmdefine format.
-FILE	Full path name of the file to contain the exported job specifications. If this parameter is not specified, the output is routed to the default output device.
-JOBNAME	Name of the job.
-MEMNAME	Member name of the job.
-FOLDER	Name of the SMART Folder or of the job. Either the JOBNAME or MEMNAME parameter is required.
-WORKING_DIR	Contains embedded script files. The name of each file consists of a SMART Folder name, job name, and time stamp. The embedded script only appears for jobs that have the in-line script turned on.

ctmexdef syntax rules

The <*name*> and <*memName*> parameters can include the following wildcard characters:

- Represents any number of characters (including none). Any parameter including * should be enclosed in quotation marks (see example below).
- Represents any single character.

To export all job processing definitions from SMART Folder PROD to file tabprod, specify the following command:

```
ctmexdef -FILE /tmp/tabprod -FOLDER PROD -JOBNAME "*"
```

ctmfw File Watcher utility

The ctmfw (Control-M File Watcher) utility monitors file status and detects the following file processes:

- Successful completion of a file transfer activity
- Creation of a file
- Deletion of a file

ctmfw can be used before activating a job or before performing a task (for example, sending a shout message or adding/deleting conditions) that is dependent upon creation or deletion of a file.

There are two usages for this utility:

- **Usage as a service:**

  As a service, ctmfw takes its parameters (rules) during startup from the **rule.dat** file whose full path name is specified in **<***Control-M/Agent***>\data\ctmfw.cfg**. For more information, see File Watcher job parameters.

- **Usage as a utility**

  When running as a utility, ctmfw is invoked from the command line. Rules can be provided on the command line or by a rule file.

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

The ctmfw utility runs as a process on a client computer. The process waits for the creation or deletion of specified files.

- For a file transfer activity, when the file is detected, the job continues to monitor the size of the file. When the file reaches a specified minimum size and does not increase in size for a specified period of time, the File Watcher utility either finishes with a status of OK or executes a specified DO action. DO actions can consist of adding or deleting conditions or executing a command.
- For file creation, file size is ignored if a wildcard is specified as part of the file name unless the mon_size_wildcard parameter is set to Y.
- For file deletion, ctmfw must first detect the existence of the file before it can detect its deletion.

The ctmfw utility can be run as follows:

- From a command line.
- Invoked to detect either a single file or multiple files. For more information, see Watching a single file (on page 192) and Watching multiple files (on page 193).
- As a job, as described in Creating a job, and using the File Watcher job parameters descriptions.

Variables can now be used in parameter fields.

All parameters must be assigned a value, even if that value is zero. If only six values are specified, the default value for mon_size_wildcard is used. If five parameters are specified, default values for wait_time and mon_size_wildcard are used, and so forth.

```
ctmfw /home/watchedfile.txt CREATE 100 10
```

is resolved by using default values for mon_int, min_detect, wait_time, and mon_size_wildcard as follows:

```
ctmfw /home/samplefile.txt CREATE 100 10 10 3 0 N
```

For more information, on the rules file, see ctmfw rule file (on page 197).

For a description of the ON_FILEWATCH parameters, see ctmfw rules file parameters (on page 198).

If an ON_FILEWATCH statement contains a cyclic_interval parameter, ctmfw will only stop monitoring a file on a DO_OK or DO_NOTOK action.

The ctmfw utility is invoked to watch multiple conditions. The definitions the ctmfw utility uses for watching each file are contained in a rule file.

Watching a single file

This procedure describes how to watch a single file with the ctmfw File Watcher utility.

➢ To watch a single file:

- Type the following command:

  ctmfw FILE (absolute path)

```
      < mode  (CREATE|DELETE)>   Default: CREATE
       < minimum detected size <number>
         ['  '|Bytes(B)|Kilo(K)|Mega(M)|Giga(G)] >Default:0
       < interval between file search (seconds) > Default: 60sec
       < interval between filesize comparison iterations (seconds) >
 Default: 10sec
       < number of iterations while the size is static >  Default: 3
 iterations
       < time limit for the process (minutes).>  Default: 0 (no time limit)
         Effective while the file does not exists or,
         the file size is static and the minimum size
         was not reached >
       <  monitor file size , minimal and maximal age, when wildcard is used
 > Default: N
       < starting  time for detecting files (HHMM or YYYYMMDDHHMM > Default:
 NOW
       < absolute stop time (HHMM or YYYYMMDDHHMM > Default: +0000 ( No stop
 time )
       < minimal age of file (modified time)
         format:xxxxYxxxxMxxxxDxxxxHxxxxMin  > Default: NO_MIN_AGE
       < maximal age of file ( timestamp monitoring )
         format:xxxxYxxxxMxxxxDxxxxHxxxxMin  > Default: NO_MAX_AGE
```

Watching multiple files

This procedure describes how to watch multiple files with the ctmfw File Watcher utility.

➢ To watch multiple files:

- Type the following command to invoke the ctmfw utility for multiple files:

  ```
  ctmfw -input <ruleFileName>
  ```

  The variable *<ruleFileName>* is the complete path name of the file containing the definitions for each file to be detected.

rule file

The following displays a sample rule file. In this sample:

- # indicates comments.
- Default values are shown for all global parameters.
- <action> refers to any of the actions described in ctmfw-valid actions below.

```
#*****************************************************************
*
# Global Parameters
INTERVAL <60>  # Sleep interval (seconds)
MIN_SIZE 4Kilo
```

```
MIN_AGE  3M24D4h5min
FROM_TIME <0000>  # Starting time for all files (hhmm)
MIN_SIZE <0>  # Minimum size for all files (bytes)
MIN_DETECT <3>  # Number of iterations for all files
WAIT_TIME <0>  # Time limit for all files (minutes)
# ON_FILEWATCH statements
ON_FILEWATCH <fileName>(absolute path) [CREATE/DELETE] [min_size]
[min_detect] [wait_time]
            [start_time] [cyclic_interval] [wildcards]
[minimal_file_age]
THEN
<action>
ELSE
<action>
END_ON
#******************************************************************
*
```

If a wildcard is used in the file name, the found file can be referenced as %FILENAME%. For example:

```
INTERVAL 10
ON_FILEWATCH /controlm/datafile*.txt CREATE
THEN
DO_COND %FILENAME% 0101 +
```

All global parameters must be delimited by the new line character.

ctmfw – valid actions

Action	Description
DO_CMD <command>	Execute a valid command under the command interpreter. Full path names are required for files.
DO_COND <condition name> <condition date> <+|->	Add (+) or delete (-) a condition.
DO_EXIT [exit code]	Terminate ctmfw with the user-defined exit code.
DO_NOTOK [exit code]	Terminate an ON_FILEWATCH statement with status NOTOK. Exit code is optional and replaces the standard return code, as described in the ctmfw-return codes table below.
DO_OK	Terminate an ON_FILEWATCH statement with status OK. If there is more than one file in the Rule file, the result displayed is that of an AND algorithm.

- If the file is detected and the size remains static within the time frame (CREATE) or the file has been deleted (DELETE), the DO commands in the THEN block are executed.
- If the file is not detected or deleted within the time frame, the statements following the ELSE block are executed.
- ctmfw terminates when all the files in the Rules file have been processed.

ctmfw parameters

The following table lists the ctmfw parameters:

Parameter	Value
Job Name	FileWatch
Mem Name	FileWatch
Run as	<control_m_user>
From Time	1900
Command line	ctmfw "\tmp\trans.dat" CREATE 100 60 10 5 180
On Statement/Code processing:	
Stmt	*
Code	COMPSTAT=0
Do Cond	file_trans_dat_ok Date: ODAT Sign: +
Stmt	*
Code	COMPSTAT=1
Do Shout	To: Control-M/EM Text: "File trans.dat did not arrive on time"

ctmfw rule file

The Rules file contains the following sections:

- Global parameters, whose default values apply to all the files in the rule file. For more information, see ctmfw rules file global parameters (on page 201).
- ON_FILEWATCH statements identifying which files to detect, specific criteria for each file, and the action to take upon detection or non-detection. Any number of ON_FILEWATCH statements can appear in a Rules file.

All keywords must be entered in uppercase

If any mandatory parameter is omitted from a Rules file, the default value for that parameter is used. Parameters entered for ON_FILEWATCH statements override the default values. If entered, they must appear in the order shown in Figure 2.

The following instructions are defined in the Rules file:

- The sleep interval between succeeding scans must be 10 seconds.
- If the ctmfw utility detects that the datafile.txt file in the **/home/controlm** directory is created in the specified time interval, then:
  - the datafile condition dated 1 January must be added.
  - the command interpreter must execute the command to move the contents of the `~<controlm_owner>/ctm_server/datafile.txt` file to `~<controlm_owner>/ctm_server/workfile.txt`.
- If the ctmfw utility detects that the datafile.txt file in the `~<controlm_owner>/controlm` directory is not created in the specified time interval, then condition datafile dated 1 January must be deleted.
- When the ctmfw utility detects that the `~<controlm_owner>/ctm_server/tempfile.txt` file is deleted, condition tempfile dated 1 January must be deleted.

```
#******************************************************************
INTERVAL 10
ON_FILEWATCH ~<controlm_owner>\ctm_server\datafile.txt CREATE
THEN
DO_COND datafile 0101 +
DO_CMD move \ctm\datafile.txt \ctm\workfile.txt
ELSE
DO_COND datafile 0101 -
END_ON
ON_FILEWATCH \ctm\tempfile.txt DELETE
THEN
DO_COND tempfile 0101 -
END_ON

#****************************************************************
```

A job processing definition is created to implement a FileWatcher job. The file must arrive between 19:00 and 22:00, and be created in the /tmp directory under the name trans.dat. The minimum file size is 100 bytes. The detection process should be performed each minute. The file size is monitored every 10 seconds, and the number of intervals where the file size remains static is 5. If the file is not detected by 22:00, an alert should be sent to Control-M/EM.

ctmfw rules file parameters

The following table lists the ctmfw utility rules file parameters:

Parameter	Description	
FILE	Path of the file to be detected. The file name can include wildcard character * to represent any number of characters (including no characters) or ? to represent any one character. The path and file name must not exceed 214 characters.	
mode	CREATE	Detects creation of a file. Default. File size is ignored if the filename parameter contains wildcards (unless the **monitor file size when wildcard is used** parameter is set to Y).
		If a mask is specified for the filename, and the monitor file size when wildcard is used parameter is set to: ■ N, the ctmfw utility will end OK after detection of the first file that matches the specified mask. ■ Y, the ctmfw utility will end OK after detection of the first file that matches the filename and file size.
		For more information about **monitor file size when wildcard is used**, see below.
	DELETE	Detects deletion of a file. When the ctmfw utility is run in this mode, it first checks for files that match the specified name. After a specified file is detected, the ctmfw utility checks at the specified interval for deletion of that file.
		If a mask is specified as the filename, the ctmfw utility will end successfully only after all detected files that match the specified mask have been deleted.

Parameter	Description
minimum detected size	Minimum file size in bytes. This parameter is ignored if the FILE parameter contains wildcards (unless the **monitor file size when wildcard is used** parameter is set to Y) or if the mode parameter is set to DELETE. Default: 0 (any size detected).
interval between file searches	Interval between successive attempts to detect the existence/deletion of a file (in seconds). Default: 60
interval between filesize comparison iterations	Interval between attempts to monitor the size of a file after it is detected (in seconds). This parameter is ignored when using wildcards in FILE or when using DELETE mode. Default: 10
number of iterations while size is static	Number of attempts to monitor file size where the size remains static and greater than or equal to **minimum detected size** (indicating successful creation of the file). This parameter is ignored when using wildcards in FILE or when using DELETE mode. Default: 3
time limit for the process	Maximum time (in minutes) to run the process without detecting the file at its minimum size (CREATE) or detecting its deletion (DELETE). If the file is not detected/deleted in this specified time frame, the process terminates with an error return code. Default: 0 (no time limit).
monitor file size when wildcard is used	Indicates whether file size should be monitored if the filename contains wildcards. This parameter is ignored if the filename does not contain a wildcard. Valid values: ▪ N – do not monitor file size (Default) ▪ Y – monitor the file size If this parameter is set to Y and more than one file matches the specified mask, the ctmfw utility selects the first file that is detected, monitors its file size, and ignores all other matching files.
starting time for detecting files	Indicates an absolute time at which the utility starts monitoring the file. For example, 200712061400, means that at 2 PM on December 6th, 2007 the FileWatcher utility will start watching the file. Alternatively, you can use the HHMM format, in which case the current date is used.
absolute stop time	Indicates an absolute time at which the file is no longer watched. For example, 200702061400, would mean that at 2 PM on February 6th, 2007 the FileWatcher utility will stop watching the file. Alternatively, you can use the HHMM format, in which case the current date is used.
maximal age of file	Indicates the maximum amount of time that can pass since the file you want to watch was last modified. For example, 2y3d5h means that after 2years, 3 days, and 5 hours has passed, the file will no longer be watched. Entering a value of 2H10Min, means that after 2 hours and 10 minutes has passed, the file will no longer be detected. This parameter is ignored if the mode parameter is set to DELETE. Default: 0

Parameter	Description
minimal age of file	Indicates the minimum amount of time that must have passed since the file you want to watch was last modified. For example, 2y3d5h means that 2years, 3 days, and 5 hours must pass before the file will be watched. Entering a value of 2H10Min, means that 2 hours and 10 minutes must pass before the file will be detected. This parameter is ignored if the mode parameter is set to DELETE. Default: 0

ctmfw rules file global parameters

The following table lists the ctmfw rules file global parameters:

Parameter	Description
CYCLIC_INTERVAL	Indicates the interval between multiple operations of detecting the file (in minutes). This interval must be greater than the value for WAIT_TIME. If the cyclic_interval is 0, only one attempt to detect the file will be performed. Default: 0
FROM_TIME	Starting time for detecting all the files (default FROM_TIME). Used with WAIT_TIME to identify the time frame for detecting and monitoring the files. This parameter is expressed in 24-hour, hhmm format. Default: 0000 or Now
INTERVAL	Sleep interval (in seconds) between successive scans for all the files. This parameter replaces individual sleep_int and mon_int parameters for each file. Default: 10
MAX_AGE	Indicates the maximum amount of time that can pass since the file you want to watch was last modified. ▪ If MAX_AGE = 0, any change to the file timestamp means that the condition is met. ▪ IF MAX_AGE = 10 Min and if the amount of time of the watched file that has passed is less than 10 minutes, then the condition is met. This parameter is ignored if the mode parameter is set to DELETE. Default: 0
MIN_AGE	Indicates the minimum amount of time that must have passed since the file you want to watch was last modified. For example, 2y3d5h means that 2years, 3 days, and 5 hours must pass before the file will be watched. This parameter is ignored if the mode parameter is set to **DELETE**. Default: 0
MIN_SIZE	Minimum file size in bytes. This parameter is ignored if the FILE parameter contains wildcards (unless the monitor file size when wildcard is used parameter is set to Y) or if the mode parameter is set to DELETE. Default: 0 (any size detected).
MON_SIZE_ WILDCARD	Indicates whether file size should be monitored if the filename contains wildcards. This parameter is ignored if the filename does not contain a wildcard. Valid values: ▪ N – do not monitor file size (Default) ▪ Y – monitor the file size If this parameter is set to Y and more than one file matches the specified mask, the ctmfw utility randomly selects one matching file, monitors its file size, and ignores all other matching files.

Parameter	Description
STOP_TIME	Indicates an absolute time at which the file is no longer watched. For example, 200702061400, means that at 2 PM on February 6th, 2007 the FileWatcher utility will stop watching the file. You can also use the HHMM format, which uses the current date, plus the HHMM entered. Default: 0 (meaning, no stop time) STOP_TIME can only be used as a global parameter.
WAIT_TIME	Maximum time (in minutes) to run the process without detecting the file at its minimum size (CREATE) or detecting its deletion (DELETE). If the file is not detected/deleted in this specified time frame, the process terminates with an error return code, as described in ctmfw- return codes table below.. Default: 0 (no time limit).

Return codes

The return codes listed in the following table are issued by the ctmfw utility after detecting if a file is created or deleted in the specified time frame.

Return code	Description
0	File was successfully created or deleted (file arrived in the specified time frame and file size is above or equal to the minimum specified size).
1	▪ Utility failed, for example, because of a syntax error. ▪ A DO_NOTOK statement occurred, but no user-defined exit code was provided for the DO_NOTOK statement.
7	Indicates that the ctmfw request timed out. That is, the file was not detected in the specified time frame.

FileWatcher silent mode registry key

The FileWatcher service does not open an additional window during execution. If you want visual feedback while running the service, the following registry key setting must be changed to N.

```
HKEY_LOCAL_MACHINE\SOFTWARE\BMC_Software\
Control-M\FileWatcher\SYSPRM\Silent_Mode
```

ctmimptb

The ctmimptb utility imports job definition folders (including SMART Folders and Rule-Based Calendars) that were exported from Control-M/EM by the exportdeffolder utility. To run the ctmimptb utility, see Running the ctmimptb utility (on page 203).

The ctmimptb utility gets the exported XML file that was created by exportdeffolder (on page 298) in Control-M/EM, and imports the folders into the Control-M/Server database. This utility can overwrite existing folders when overwrite mode is specified.

The ctmimptb utility works at the folder level only and cannot import a specific job or jobs from a folder. The utility can import Sub-folder entities and Sub-folders jobs, but cannot import Sub-folders individually. The Sub-folders must be imported as part of SMART folders.

The Control-M/EM exportdeffolder utility cannot define an empty folder. Therefore, the ctmimptb utility cannot perform an action that will cause folders to become empty. If you duplicate SMART Folders in the XML input file, when overwrite mode is specified, the last SMART Folder is considered and overwrite mode is not specified and the SMART Folder does not exist in the database, the first defined SMART Folder is considered.

Running the ctmimptb utility

This procedure describes how to run the ctmimptb utility, which enables you to import job definition folders (including SMART Folders and Rule-Based Calendars) that were exported from Control-M/EM by the exportdeffolder utility.

To run the ctmimptb utility:

- Type the following command:

```
ctmimptb
  -PATH      <full path to the output XML file exported by exportdeffolder EM utility>
  [ -OVERWRITE ]
  [ -DEBUG  <debug level 0-5 ]
  [ -MODULE  <debug module 0-3 ]
  [ -H | -Help ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Description of the import procedure

When a job definition folder in the input XML file is large, importing this folder is performed by a few transactions. The number of jobs within each transaction is specified by the CTMIMPTB_ENT_IN_TRANS parameter that is defined in the **~controlm/ctm_server/data/config.dat** file. If the CTMIMPTB_ENT_IN_TRANS parameter is not specified in the **config.dat** file. The value range is 5–10,000 jobs and the default value 1,000 jobs.

The following describes the results of the import procedure:

- In the event of a successful import, the utility exits with a success return value. If the utility was invoked without overwrite mode and the utility encounters rejected folders during the import process, an appropriate message is issued for the partial import.
- In the event of a failure, the utility exits with an error return value and a partial import might occur. If a commit was already made, an appropriate message is issued for the partial import.

The ctmimptb utility is run by using the Control-M/Server security authorization of the invoking user.

The permissions that are required to run the ctmimptb utility on Control-M/Server SMART Folders are Read, Write, and Delete.

If permission is denied for a specific SMART Folder, the ctmimptb utility ignores the folder, continues processing the input XML file, and (if no other errors are encountered) exits with a success return value.

BMC recommends that you avoid updating job definition folders while the ctmimptb utility is running.

ctmimptb parameters

The following table lists the ctmpimptb utility parameters:

Parameter	Description
-PATH	indicates the full path to the file that holds the job definition folders to be imported This file is in XML and is generated by the exportdeffolder utility in Control-M/EM.
-OVERWRITE	determines how the utility responds when a folder that already exists in Control-M/Server is imported: ▪ When this parameter is specified, the current folder in the Control-M/Server database is replaced with the imported folder, and a message similar to the following message is issued: "Schedule folder already exists in the database and overwrite was specified. Overwriting this folder" ▪ When this parameter is not specified, the imported folder is rejected, leaving the current folder in the Control-M/Server database unchanged, and a message similar to the following message is issued: "Schedule folder already exists in the database and overwrite was not specified. Skip importing this folder." After either using the imported folder to replace the current folder in the Control-M/Server database or rejecting the imported folder, the ctmimptb utility continues processing the file.
-DEBUG	activates a debug trace at the specified level Valid levels are 0-5. The default is **0**. Performance is slower when Control-M/Server is operating in debug mode. BMC recommends that you activate debug mode only when requested by Customer Support.
-MODULE	indicates which components are to be traced for diagnostic purposes Valid values are as follows: ▪ 0 – all components ▪ 1 – common functionality flow (default) ▪ 2 – event manager ▪ 3 – database layer
-H | -Help	displays the usage of the utility

ctmkilljob

The ctmkilljob utility terminates a specified Control-M job and all its processes. ctmkilljob terminates only jobs that are currently executing. This utility can only be run interactively. To run the utility, see Running the ctmkilljob utility (on page 206).

The ctmkilljob utility fails if the agent is upgrading.

The ctmkilljob utility can be invoked using the -input_file parameter:

```
ctmkilljob -input_file <fullPathFileName>
```

The referenced file contains all the required parameters. Most of the parameters are described in Control-M Parameters andctmkilljob parameters (on page 207).

If the action performed by the ctmkilljob utility was successful, the utility responds with the statement:

```
Job was killed
```

The specified job is ended with NOTOK status.

The parameters specified for ctmkilljob must indicate one unique job. If more than one job fits the description specified in the command, you are informed that a unique name must be entered to carry out the action. Reenter the command with parameters that specify one unique job.

Running the ctmkilljob utility

This procedure describes how to run the ctmkilljob utility, which enables you to terminate a specified Control-M job and all its processes.

➢ To run the ctmkilljob utility:

- Specify one of the following commands to invoke the ctmkilljob utility:

  -
    ```
    ctmkilljob
    [ -ORDERID      <uniqueOrderID> ]
    [ -HOSTID       <hostID> ]
    [ -MEMLIB       <path> ]
    [ -MEMNAME      <fileName> ]
    [ -JOBNAME      <jobName> ]
    ```

  -
    ```
    ctmkilljob -input_file <fullPathFileName>
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmkilljob parameters, see ctmkilljob parameters (on page 207).

ctmkilljob parameters

The following table lists the ctmkilljob utility parameters:

Parameter	Description
-ORDERID	Control-M Order ID of the job to be terminated.
-HOSTID	Host name of an agent computer, or name of a host group to which the job should be submitted.
-MEMLIB	Name of the library/directory in which the job script resides.
-MEMNAME	Name of the file that contains the job script statements.
-JOBNAME	Descriptive reference for a job processing definition.
input_file	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: prepare and save files of utility parameters that can be reused. specify utility input longer than the number of characters allowed in the command line. `-input_file ~<controlm_owner>/ctm_server/data/ctmkilljob_parms.txt`

ctmorder

The ctmorder utility orders or forces one or more jobs from a SMART Folder in the Control-M/Server database:

- Ordered jobs are placed in the Active Jobs database if their scheduling criteria are met.
- Forced jobs are placed in the Active Jobs database regardless of their scheduling criteria.

To run the ctmorder utility, see Running the ctmorder utility (on page 208).

If two jobs with the same name exist in a SMART Folder and you use the ctmorder utility to order or force a job with that name, only the first job is added to the Active Jobs database.

If the ctmorder utility is running when the New Day procedure begins, it is automatically suspended until New Day procedure is ended.

When ordering a SMART Folder, folder-level Rule-Based Calendars are calculated and joined so that if the scheduling criteria are met, the folder will be ordered. As a result of ordering the folder:

- A row in the Active Jobs database is added for this folder.
- All folder contents must pass the order procedure. Each field in the folder is inspected as follows:
  - For regular jobs, the job scheduling criteria is calculated and either Or'ed (default) or And'ed with folder or Sub-folder-level Rule-Based Calendars associated to it, according to the relationship parameter in the job definition. If the scheduling criteria are satisfied the job is inserted into the Active Jobs database. If the scheduling criteria are not satisfied, the job is ignored.
  - For Sub-folders, Rule-Based Calendars of the Sub-folder are calculated and joined. When * is defined, Rule-Based Calendars of the parent folder are fetched. If the result of the Rule-Based Calendars calculation is satisfied, a row for the Sub-folder is added in the Active Jobs database and the Sub-folder content will be ordered. If the result of the Rule-Based Calendars calculation is not satisfied, the Sub-folder is ignored.
- Ordered SMART Folder, jobs and Sub-folders status are set to WAIT_SCHEDULING.
- INTO_FOLDER_ORDERID will be used to force (Ordering Sub-folders will not be valid) a job or Sub-folder into already ordered folder or Sub-folder. The ordered job or Sub-folder should belong to the same folder or Sub-folder of the job or Sub-folder we are ordering into. Force a Sub-folder ALONE will not be applicable, for jobs it will be applicable.

The ctmorder utility can be invoked using the -input_file parameter:

```
ctmorder -input_file <fullPathFileName>
```

The referenced file contains all the required parameters. Most of the parameters are described in Control-M Parameters and ctmorder parameters (on page 209).

Running the ctmorder utility

This procedure describes how to run the ctmorder utility, which enables you to order or force one or more jobs from a SMART Folder in the Control-M/Server database.

➢ To run the ctmorder utility:

- Type one of the following commands:
  - The first format contains only a few parameters in a specific order:

```
ctmorder <SMART Folder name> <jobName> <odate>\ [{order|force}]
```

This first format cannot be used if the ctmorder utility is invoked from a Control-M/Agent computer.

- The second format allows specification of all optional parameters in any order but requires each specified parameter to be named. Format:

```
ctmorder -FOLDER <Folder|SMART Folder|Folder Path> -NAME    <job
name|sub folder name> -ODATE <scheduling date>
[-FORCE <y|n>]
[-HOLD  <y|n>]
[-UNIQUE  <y|n>]
[-SEQNO <job sequence number>]
[-INTO_FOLDER_ORDERID <{SMART folder order id}|LAST|ALONE|NEWT]>
[-NODUPLICATION]
[-DEBUG    <debug level 0-5> ]
[-QUIET  ]
[-VARIABLE <varname> <expression> ]
[-ODATE_OPTION <VALUE_DATE|RUN_DATE>]
-or-
ctmorder -input_file <fullPathToFileName>
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmorder parameters, see ctmorder parameters (on page 209) and ctmorder example (on page 213).

ctmorder parameters

The following table lists the ctmorder utility parameters:

Parameter	Description	
-FOLDER	Either a folder name, SMART folder name (name of the SMART Folder containing the jobs) or the path (including the folder name) to the folder. This value might contain wild characters within the folder path. Wild characters in the outermost folder name are also valid.	
-UNIQUE	Adds a unique suffix to every condition name.	
	Y	Add a unique suffix to every condition name.
	N	Do not add a unique suffix to every condition name. Default

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-NAME</td><td colspan="2">Job name (or mask) of the job(s) to order or force. Mandatory.
You can order a SMART Folder, but you cannot order an individual job, or selection of jobs, from a SMART Folder.
The job name can include the following wildcard characters:
▪ * — represents any number of characters (or none). Specify * by itself to include all jobs in the folder.
Any parameter including * must be enclosed in double quotation (see Example below).
▪ ? — represents any single character.
▪ $ — represents any single character.
Whether the *, ?, and $ characters act as wildcards or as ordinary characters in a job name is determined by the presence or absence of a -seqno parameter:
The *, ?, and $ characters will only act as wildcards in the -jobname parameter if no -seqno parameter is specified.
The *, ?, and $ characters will only act as ordinary characters in the -jobname parameter if a -seqno parameter is specified. In this case, the specified job name must exactly match the name of the job in the indicated sequence, or the job will not be ordered.</td></tr>
<tr><td rowspan="4">-odate</td><td colspan="2">Indicates the scheduling date (odate) to be associated with job(s). Valid values:</td></tr>
<tr><td>ODAT</td><td>The current working date of the computer on which Control-M/Server is running.
This is the default value.</td></tr>
<tr><td>yyyymmdd</td><td>A specific working day in yyyymmdd format.</td></tr>
<tr><td colspan="2">The interpretation of this parameter value depends on the value specified for the -odate_option parameter (described below).</td></tr>
<tr><td rowspan="3">-force</td><td colspan="2">Add the specified jobs to the Active Jobs database regardless of scheduling criteria. If -force is not specified, jobs are added to the Active Jobs database only if their scheduling criteria are satisfied (known as order).
Use -INTO_FOLDER_ORDERID to force a job or sub-folder into a folder or sub-folder that has already been ordered.</td></tr>
<tr><td>Y</td><td>Force the specified jobs.</td></tr>
<tr><td>N</td><td>Order the specified jobs. Default.</td></tr>
</table>

Parameter	Description	
-hold	Place the specified jobs in the Active Jobs database in Hold status.	
	Y	Hold the specified jobs.
	N	The specified jobs are not held. Default.
-seqno	A counter identifying the row number of the job in the SMART Folder. The first job in each SMART Folder is numbered 1 and each subsequent job increments the counter by one. If this parameter is not specified, the first job in the specified folder is ordered. Optional. If this parameter is specified, any *, ?, and $ characters in the -jobname parameter are assumed to be ordinary characters rather than wildcards. Therefore: Do not specify a -seqno parameter if you are specifying wildcards in the -jobname parameter. If you are specifying *, ?, or $ characters as ordinary characters (not wildcards) in the -jobname, you must specify the appropriate -seqno parameter (and the specified job name must exactly match the actual job name).	
-INTO_FOLDER_ ORDERID	This parameter is relevant only for jobs in a SMART Folder. If the folder being ordered is not a SMART Folder, this parameter is ignored.	
	SMART Folder order id	Order ID of an existing SMART Folder.
	LAST	Jobs are added to the last ordered instance of their SMART Folder in the Active Jobs database.
	ALONE	Jobs are ordered individually. They are not associated with any SMART Folder.
	NEWT	A new folder is created and the specified jobs are ordered to that SMART Folder. Default.

Parameter	Description	
	The SMART Folder order id, last, alone and newt options can only be used when the -force parameter is set to **Y**.	
-noduplication	Allow jobs to be ordered and added to an existing ordered SMART Folder only if those jobs have not already been ordered in that instance of the SMART Folder. This parameter can be specified only if last or <SMART Folder order id> is specified for the -INTO_FOLDER_ORDERID parameter. This parameter is relevant only for jobs in a SMART Folder.	
-debug level	Activates a debug trace at the specified level. Valid levels: 0 – 5. Default: 0 Performance is somewhat slower when operating in debug mode. BMC recommends that you activate debug mode only when requested by Technical Support.	
-quiet	Suppresses display of the utility output. If specified, no information messages are displayed during execution of the command.	
-variable	Adds an Variable expression to each job, SMART Folder, or SMART Folder that is ordered by the utility. For more information, see Control-M Variable facility . The following information must be specified for each new variable.	
	<varName>	Name of the variable.
	<expression >	Value assigned to the variable.
-odate_option	Indicates how the specified -odate value should be used. Valid values, specify one of the following:	
	value_date	The specified odate is the odate value for the job(s). The jobs should be run during the current working day. Default. If a time zone is specified in a job processing definition, the job is run according to the specified time zone.

Parameter	Description	
	run_date	Jobs ordered by this run of the ctmorder utility should be run only when the specified odate begins. If the specified odate is the current working day, the job will run as described in value_date above. If the specified odate has not begun (for example, due to time zone specifications), then the job will wait in the Active Jobs database (with WAIT_ODAT status) until the start of the specified working day. If the specified odate has already passed, the ctmorder utility will not run and an error message will be displayed.
-input_file	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: prepare and save files of utility parameters that can be reused. specify utility input longer than the number of characters allowed in the command line. `-input_file ~<controlm_owner>/ctm_server/data/ctmorder_parms.txt`	

If neither ORDER nor FORCE is included in the command, the specified jobs are ordered.

ctmorder example

The following are example usages of ctmorder utility:

The following command orders jobs named acct_job contained in SMART Folder ACCT100. Any jobs placed in the Active Jobs database will have the current Control-M date as their original scheduling date:

```
ctmorder -FOLDER ACCT100 -NAME acct_job -ODATE odat
```

The same results can be achieved using the -input_file parameter as follows:

```
ctmorder -input_file
~<controlm_owner>/ctm_server/data/ctmorder_acct100.txt
```

The referenced file contains the following lines:

- `-FOLDER ACCT100`
- `-NAME acct_job`
- `-ODATE odat`

The following command orders all jobs contained in the SMART Folder ACCT100 whose job name begins with ga. Any jobs placed in the Active Jobs database will have the date March 15, 2010 as their original scheduling date:

```
ctmorder -FOLDER ACCT100 -NAME "ga*" \
-ODATE 20100315 -FORCE y
```

The following command forces all jobs contained in the ACCT101 sub-folder. Any jobs placed in the Active Jobs database will have the date December 31, 2009 as their original scheduling date:

```
ctmorder -FOLDER ACCT100 -NAME ACCT101 \
-ODATE 20091231 -FORCE y
```

The following command forces the third job contained in the SMART Folder ACCT200 whose job name parameter consists of prodyjob. This job is placed in the Active Jobs database and will have the date December 31, 2009 as its original scheduling date. This job is added to a folder whose orderid is B2.

```
ctmorder -FOLDER ACCT200 -NAME prodyjob \
-ODATE 20091231 -FORCE y -SEQNO 3 -INTO_FOLDER_ORDERID B2
```

The following command assigns the variable %%PRDNDATE with the value of %%ODATE, and orders every job in the PRODUCTION SMART Folder whose job name has a prefix of PRDN. These jobs are placed in the Active Jobs database in a folder and are assigned the date December 31, 2009 as their original scheduling date.

```
ctmorder -FOLDER PRODUCTION -NAME "PRDN*" \
-ODATE 20091231 -ORDER y -INTO_FOLDER_ORDERID newt\

-VARIABLE %%PRDNDATE %%ODATE
```

The following command orders every job in the INVENTORY SMART Folder whose job name has a prefix in the range BIN_A1 to BIN_A9. These jobs are placed in the Active Jobs database in a new SMART Folder, and are assigned December 31, 2009 as their original scheduling date. The APPLICATION and OWNER parameters of these jobs are modified to STOCK_COUNT and STOREMAN, respectively.

```
ctmorder -FOLDER INVENTORY -NAME "BIN_A?"  \
-ODATE 20091231 -FORCE n -INTO_FOLDER_ORDERID newt   \
-VARIABLE %%PRDNDATE %%ODATE  \
-VARIABLE %%APPLIC STOCK_COUNT  \
-VARIABLE %%OWNER STOREMAN
```

ctmpsm

The ctmpsm utility can be invoked interactively to display the Control-M Production Support menu. This menu is used to perform functions affecting jobs or conditions in the active jobs database of the data center. It provides an alternative to using the Control-M Workload Automation and enables you to perform many of the GUI functions directly in the data center. To run the ctmpsm utility, see Running the ctmpsm utility interactively (on page 215). For command line invocation, see ctmpsm utility command line parameters (on page 227).

The functions in this menu are divided into the following categories:

- Active Jobs database functions provide various views of the Active Jobs database. Each view displays information about the jobs and provides options to perform such actions on the jobs as Hold, Free, Delete, Rerun, Why, Confirm, View or modify job details, and view the Control-M log.
- Resource Map functions enable you to view and modify Quantitative resources, Control resources, and prerequisite conditions. The first three of these functions activate the ecactltb, ecaqrtab, and ctmcontb utilities respectively.
- Scheduling Functions enable you to order or force SMART Folders or specific jobs in SMART Folders. You can also generate monthly or yearly scheduling plans using the ctmrpln utility.

- If long names have been used for the In condition, jobname, override path, memlib, and doclib parameters, these values will be truncated in the output of the ctmpsm utility. To view the complete values for these parameters, use Control-M Workload Automation.
- The following special characters are disabled when they occur in prerequisite condition names:
  - (open parenthesis
  -) close parenthesis
  - | vertical bar
  - space

All Active Jobs database options display the following menu at the bottom of the screen:

```
H) Hold, F) Free, D) Delete, U) Undelete, R) Rerun, W) Why, Z) Details
LO) LogOrd, LJ) LogJob, Cn) Confirm, Sx)Sort[x: 1.ORDERNO 2.JOBNAME]
J) Output A) Statistic V) View Script/JCL  K) Force OK I)Dependencies JobsGx)
Global action x [x: H(Hold), F(Free), D(Delete), U(Undelete), R(Rerun)]
Q)Quit
```

```
Enter Option:
```

These actions are described in ctmpsm active jobs database actions (on page 219).

Running the ctmpsm utility interactively

This procedure describes how to run the ctmpsm utility, which enables you to display the Control-M Production Support menu.

➢ To run the ctmpsm utility interactively:

1. Log on to the server computer as the Control-M for Databases owner (for example, user controlm).

2. Specify the following command at the command prompt:

   ```
   ctmpsm
   ```

   The following menu is displayed:

   ```
   +-------------------------------+
   |   Production Support Menu     |
   +-------------------------------+
    Active Jobs Database                        Resource Map
    ----------------                        ------------
   1) List All                             61) Control Resources
   2) List All (Show Started/Ended)        62) Quantitative Resources
   3) List All (Show Application)          63) Prerequisite Conditions
   4) List All (Show Mem Name)             64) Control Resources Usage
    65) Quantitative Resources Usage
    5) List Jobs That Ended OK
    6) List Jobs That Ended NOTOK
    7) List Submitted/Executing Jobs
   8) List Cyclic Jobs
   9) List Jobs Waiting for Time Window
   10) List Jobs Waiting for Confirmation
    Scheduling Functions
    --------------------
   40) List Application/Group Tree          71) Folders
   41) List Folders                          72) Order Folders/Jobs
   42) List Ordered SMART Folders
   Q) Quit
   Enter Option:
   ```

The ctmpsm utility can also be invoked by the Command Line Interface, as described in cli (on page 303).

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmpsm options and actions, see:

- ctmpsm active jobs database options (on page 218)
- ctmpsm active jobs database actions (on page 219)
- Resource map options (on page 221)
- Scheduling function options (on page 221)
- ctmpsm folder option output example (on page 222)
- ctmpsm options for scheduling functions (on page 223)
- ctmpsm list jobs # option output example (on page 225)
- ctmpsm options in the SMART Folder List Jobs menu (on page 225)
- Values in STATE and STATUS ctmpsm fields (on page 226)

ctmpsm active jobs database options

The following table lists options that are used to perform various tasks using information in the Active Jobs database:

Code	Option	Description
1	List All	Lists all jobs in the Active Jobs database and indicates if they are associated with a SMART Folder (TBL) or Sub-folder (STB).
2	List All (Show Started/Ended)	Lists all jobs in the Active Jobs database. Indicates which jobs have started or ended execution.
3	List All (Show Application)	Lists all jobs in the Active Jobs database. Indicates the application to which each job belongs.
4	List All (Show Mem Name)	Lists all jobs in the Active Jobs database. Indicates the Mem Name for each job.
5	List Jobs That Ended OK	Lists jobs in the Active Jobs database with a completion Ended OK status.
6	List Jobs That Ended NOTOK	Lists jobs in the Active Jobs database that have a completion Ended NOTOK status.
7	List Submitted/ Executing Jobs	Lists jobs in the Active Jobs database that are currently executing.
8	List Cyclic Jobs	Lists jobs in the Active Jobs database that are cyclic.
9	List Jobs Waiting for Time Window	Lists jobs in the Active Jobs database that are waiting to begin executing based on their Time From parameter.
10	List Jobs Waiting for Confirmation	Lists jobs in the Active Jobs database that are waiting for confirmation.
40	List Application/Sub-application Tree	Displays a summarized list of the applications and sub-applications for all jobs currently contained in the Active Jobs database.
41	List SMART Folders	Displays a list of all SMART Folders in the Active Jobs database.
42	List Ordered Sub-applications	Displays a list of all Control-M sub-applications in the Active Jobs database.

ctmpsm active jobs database actions

The following table lists actions you can perform on ctmpsm active jobs database options (on page 218):

Option	Action	Description
H	Hold	Hold a job.
F	Free	Free a previously held job.
D	Delete	Mark a job for deletion. Jobs with Executing or Submitted status cannot be deleted.
U	Undelete	Undelete a job marked for deletion.
R	Rerun	Rerun a job. Rerun actions cannot be performed on a group entity.
W	Why	Display why a job has not yet been submitted.
Z	Details	View or modify (zoom and save) a job’s parameters. When a job is being viewed, it is automatically held. After changes are made and saved, the job is freed. If prerequisite conditions are added to or deleted from a job in the Active Jobs database using the Z option, the changes are automatically saved when you quit. If a cyclic job is terminated by a Do Stop Cyclic, -dostopcyclic, or DO ACTION="SPCYC" parameter, the view will contain Cyclic:T where T indicates Terminated.
LO	LogOrd	List Control-M log entries for a specific Order ID.
LJ	LogJob	List Control-M log entries for a specific Job Name.
Cn	Confirm	Confirm submission of a job.
S1	Sort (by Order No.)	Sort jobs displayed by Order number.
S2	Sort (by Job Name)	Sort jobs displayed by Job Name.
J	Output	Display the job OUTPUT.
A	Statistic	Display the job statistics

Option	Action	Description
V	View Script/JCL	View a job’s script or JCL. This option is available if Control-M for Databases is active.
K	Set to OK	Set the status of a job to OK.
I	Dependencies Jobs	Display all jobs that depend on the specified job.
GH	Global Action (Hold)	Hold all jobs in the displayed list.
GF	Global Action (Free)	Free all jobs in the displayed list.
GD	Global Action (Delete)	Mark all jobs in the displayed list for deletion.
GU	Global Action (Undelete)	Undelete all jobs marked for deletion.
GR	Global Action (Rerun)	Rerun all jobs in the displayed list. Rerun actions cannot be performed on a group entity.
The R (Rerun) option and the Global options (GH, GF, GD, GU, and GR) affect only jobs and not SMART Folders.		

Resource map options

The following table lists resource map options for the ctmpsm utility:

Code	Option	Description
61	Control Resources	Lists Control resources currently used in the Active jobs database. Activates the ecactltb utility.
62	Quantitative Resources	Allows you to list, add, modify, or delete Quantitative resources in the Active jobs database. Activates the ecaqrtab utility.
63	Prerequisite Conditions	Allows you to view, add or delete prerequisite conditions in the Active jobs database. Activates the ctmcontb utility.
64	Control Resources Usage	Shows current usage of Control resources by jobs in the Active Jobs database.
65	Quantitative Resources Usage	Shows current usage of Quantitative resources by jobs in the Active Jobs database.

Scheduling function options

The following table lists scheduling function options for the ctmpsm utility:

Code	Option	Description
71	Folders	Lists SMART Folders and jobs defined in the Control-M/Server database. Allows you to force SMART Folders or jobs, add or delete SMART Folders and generate scheduling reports.
72	Order Folders/Jobs	Allows you to order SMART Folders or jobs. You are prompted to specify: ▪ SMART Folder ▪ Job Name (optional) ▪ Odate (YYYYMMDD/ODAT) ▪ Odate_option (VALUE_DATE|RUN_DATE) [VALUE_DATE] ▪ Hold Option (Y|N) For more information about ordering jobs and SMART Folders, see ctmorder (on page 208)

ctmpsm folder option output example

When the Folders option is selected, output similar to the following is displayed:

```
Folders
 ----------------------
 Folder name           Daily name                              Folder type
1)  tab_1                                                        SMART
FOLDER
2)  temp                                                         REGULAR
3)  inventory                                                    SMART
FOLDER
4)  Payroll              Monthly
5)  inventory            SYSTEM                                  SMART
FOLDER
6)  RE_OUTPUT            output                                  REGULAR
D#) Delete UserDaily Folder #          U#) Update folder  #
F#) Force folder            #          J#) List jobs    #
A)  Add.          R)  Remove Folder    Q)  Quit.
Option []:
```

ctmpsm options for scheduling functions

The following table lists options for the scheduling functions:

If a folder that is associated with more than one Order method is modified using Control-M/EM and then uploaded to Control-M/Server, that folder is removed from all User dailies except the one that is associated with it in Control-M/EM.

Code	Option	Description
A	Add	Adds a Order method to an existed folder. When selected, you are prompted for the Folder name and Order method name.
D#	Delete UserDaily Folder #	Removes an instance of a Folder from the Control-M/Server database. If the specified instance is the only instance of the folder (that is, that folder is ordered by only one order method), the Folder and all its associated jobs are deleted. If the specified instance is not the only instance of the folder, then only the specified instance is removed from the Control-M/Server database.

Code	Option	Description
F#	Force folder #	Forces a specific Folder (for example, specify F6 to force folder RE_OUTPUT). The following prompt is displayed: Odate (YYYYMMDD/ODAT) [ODAT]: Enter the odate for the job to be forced in YYYYMMDD format, or enter the value ODAT to indicate that the job should use the current working date as its odate. The following prompt is displayed: ODATE option (VALUE_DATE|RUN_DATE) [VALUE_DATE]: To run jobs now with the specified odate, specify, VALUE_DATE To wait until the specified odate begins before running the jobs, specify RUN_DATE. If the specified folder is a SMART Folder, the following prompt is displayed: Please choose one of the following: A) Alone. N) New sub application. L) Last. B) Bind to existing group Orderno. These options are described below: ▪ A – Forces each job in the folder separately as a non-sub application job. ▪ N – Forces the jobs in the folder as a new sub application in the Active Jobs database. ▪ L – Forces the jobs in the folder, and adds them to the most recently ordered sub application in the Active Jobs database. ▪ B – Forces the jobs in the folder, and adds them to a specified application in the Active Jobs database.
J#	List jobs #	Lists content of a folder and provides options to force a specific job or sub-folder or generate a report (for example, specify J1 to list the jobs in folder supply).
R	Remove	Deletes a specific Folder and all its associated jobs (for example, specify R RE_OUTPUT to delete folder RE_OUTPUT).
U#	Update folder #	Updates the Order method name for a specific Folder (for example, specify U6 to update the Order method name for folder RE_OUTPUT).

ctmpsm list jobs # option output example

When the List Jobs # option is selected, output similar to the following is displayed:

```
Table RE_SHED Jobs
-----------------------------
1)  Jobname:DAYS_CAL_N, Memname:DAYS_CAL_NONE
2)  Jobname:DAYS_30_FE, Memname:DAYS_30_FEB
3)  Jobname:DAYS_28_29, Memname:DAYS_28_29_FEB
4)  Jobname:NO_CALENDA, Memname:NO_CALENDAR
5)  Jobname:DATES_0101, Memname:DATES_0101_0202
6)  Jobname:DATES_2902, Memname:DATES_2902
7)  Jobname:DAYS_CAL_M, Memname:DAYS_CAL_MINUS
 8)  Jobname:DAYS_CAL_P, Memname:DAYS_CAL_PLUS
9)  Jobname:DAYS_CAL_W, Memname:DAYS_CAL_WITHOUT
10)  Jobname:CALENDAR_O, Memname:CALENDAR_ONLY
11)  Jobname:wdays_all , Memname:WDAYS_ALL
12)  Jobname:wdays_1_2_, Memname:WEEKDAYS_1_2_3
Q) Quit.    F#) Force job #
M#) Month Schedule Plan # Y#) Year Schedule Plan for job #
Option []:
```

c

ctmpsm options in the SMART Folder List Jobs menu

The following table lists options in the SMART Folder List Jobs menu:

Code	Option	Description
F#	Force job #	Forces a specific job (for example, specify F2 to force job DAYS_30_FEB).
M#	Month Schedule Plan	Generates a monthly Job Order report for the folder. You are prompted to enter the year and month in format YYYYMM.
Y#	Year Schedule Plan for job #	Generates a yearly Job Order report for a specific job. You are prompted to enter the year in format **YYYY**.

Values in STATE and STATUS ctmpsm fields

The following table displays the values that are listed in the STATE and STATUS fields when ctmpsm is executed:

Value	Description
STATUSES	
OK	The job completed okay.
NOTOK	The job did not complete okay.
STATES	
Wait Sche	The job is waiting to be scheduled.
Wait Conf	The job is waiting for user confirmation.
Wait Reru	The job is waiting to be rerun.
Wait Time	The job is waiting for its time frame.
Wait Cond	The job is waiting for a condition.
Wait Reso	The job is waiting for a resource.
Wait Host	The host(s) to which the job is being submitted is unavailable, either because of host's restriction, or because of network availability.
Wait Workload	One or more of the workloads with which the job is associated has reached its maximum jobs limit policy.
Submitted	The job was submitted (that is, the job was sent to an agent).
Retry Sub	The job is waiting for a submission retry.
Executing	The job is executing.
Ended	The job has ended.
Analyzed	The job is being analyzed.
Disappear	The job has disappeared in the agent.
Post proc	The job has performed its post processing activities.
Wait ODAT	The job is waiting for the appropriate ODAT.

Value	Description
Post ODAT	The appropriate ODAT of the job already passed.
Unknown	The status of the job is unknown.

ctmpsm utility command line parameters

The following table lists the valid values for each mode of the ctmpsm utility command line interface:

Mode	Description	
CHILD	Lists dependent jobs with IN conditions that are created by the job whose order ID is specified in this command. ctmpsm -CHILD <order_ID> [<what>]	
	order_ID	Identifies the "parent" job.

	what	B=batch job
		D=detached
		C=command
		U=dummy
		X=external job

Mode	Description

IMPORT_CAL	Imports a calendar from the Control-M/EM.
	ctmpsm -IMPORT_CAL *<Control-M/EMexportedFileName>* **[overwrite]**
	<Control-M/EMexportedFileName> is the full path name of the calendars file to be imported from Control-M/EM.
	The file must be imported from Control-M/EM in xml mode only.
	If overwrite is specified, when the specified calendar to be imported from Control-M/EM already exists in the Control-M/Server database, the import action will overwrite it. Default: overwrite is not specified.
	There is a line of output for every calendar handled by the import_cal option.
	After a successful import of a calendar, the following message is displayed:
	`Calendar <x>, for year <y>, has been imported.`
	Assume calendar *<x>*, that is being imported, already exists in the Control-M/Server database, and that the overwrite option has not been specified. The following message is displayed:
	`Calendar <x>, for year <y>, that already exists, has not been imported.`

<table>
<tr><th>Mode</th><th colspan="2">Description</th></tr>
<tr><td>FULLUPDATE</td><td colspan="2">Modifies the specified parameters in the job whose order ID is specified in this command:
ctmpsm -FULLUPDATE <orderId>
  [-SUB_APPLICATION   <sub_applicationName>]
  [-APPLICATION   <applicName>]
  [-HOSTGRP   <name>]
  [-MEMLIB   <path>]
  [-MEMNAME   <fileName>]
  [-CMDLINE   <string>]
  [-RUN_AS   <userName>]
  [-MAXRERUN   <value>]
 [-TIMEFROM   <earliestSubmissionTime>]
  [-TIMEUNTIL   <latestSubmissionTime>]
  [-TIMEZONE   <timeZoneName>]
  [-PRIORITY   <jobPriority>]
  [-CRITICAL   Y|N]
  [-CYCLIC   Y|N]
  [-INTERVAL
           <45d(days)|1080h(hours)|64800m(minutes)>]
  [-OVERRIDE_PATH   <alternativeDirectory>]
  [-MAXWAIT   <days>]
  [-INCOND   <condition> <dateRef>|ODAT AND|OR]
  [-OUTCOND   <condition> <dateRef>|ODAT ADD|DEL]
  [-VARIABLE   <varName> <expression&gt;]</td></tr>
<tr><td rowspan="3">LISTCAL</td><td colspan="2">Lists available calendars. The list can be restricted by calendar name and year.
ctmpsm -LISTCAL [<em>&lt;calName&gt;</em>][<em>&lt;calYear&gt;</em>]</td></tr>
<tr><td><em>&lt;calName&gt;</em></td><td>Restricts the list to calendars with the specified name or prefix (indicated by * at the end).</td></tr>
<tr><td><em>&lt;calYear&gt;</em></td><td>Restricts the list to calendars for the specified year.</td></tr>
</table>

Mode	Description
LISTALL	Lists jobs in the Active Jobs database. The list can be filtered by time, application, and member name. The list can be sorted by order ID or job name. ctmpsm -LISTALL [{ODAT|TIME|APPLICATION|MEMNAME|ALL|ALLFIELDS}] [-SORT {ORDERID|JOBNAME}] In addition to the order ID and the job name, one of the following fields must also be included in the LISTALL output: ▪ ODAT – Order date. ▪ TIME _ Time execution started and ended. ▪ APPLICATION _ Application to which the job belongs. ▪ MEMNAME _ Member name for the job. ▪ ALL – Includes ODAT, FROMTIME and UNTIL fields. ▪ ALLFIELDS – Includes ODAT, MEMNAME, and APPLICATION fields. ▪ –SORT indicates the order in which the jobs should be listed. Valid values: ORDERID and JOBNAME. Note for other job statuses: The following additional job statuses are visible only when using the LISTALL option: ▪ WAIT_ODAT The Job’s Odate is later than the Control-M/Server working date for the relevant timezone. The job is waiting for the relevant day (Odate) to begin. ▪ POST_ODAT The job’s Odate is earlier than the Control-M/Server working date for the relevant timezone. The job will be deleted during the next run of the New Day procedure. These job statuses are used in the Control-M/Server Active Jobs database. However, in Control-M/EM, jobs with these statuses will appear with Wait Condition status. In non-interactive mode, WAIT_Condition and WAIT_CONFIRM are both displayed as Wait Con.

Mode	Description
LISTJOB	Lists jobs that are cyclic, as well as jobs in the Active Jobs database with a specified status. Jobs can be filtered by status: OK, NOTOK, executing, waiting for the end of a time interval, waiting for confirmation. ctmpsm -LISTJOB {OK|NOTOK|EXECUTING|CYCLIC| WAITTIME|WAITCONFIRM} [-SORT {ORDERID|JOBNAME}] ■ OK _ Jobs with a completion Ended OK status. ■ NOTOK _ Jobs with a completion Ended NOTOK status. ■ EXECUTING _ Jobs that are currently executing. ■ CYCLIC _ Jobs that are cyclic. ■ WAITTIME _ Jobs waiting to begin executing based on the time specified in their Time From parameter. ■ WAITCONFIRM _ Jobs waiting for confirmation. ■ –SORT indicates the order in which the jobs should be listed. Valid values: ORDERID and JOBNAME.
LISTSUB_APPLICATION	Lists jobs in the specified sub-application that are associated with a specified application. ctmpsm -LISTSUB_APPLICATION <application> <sub_application>> [<scheduling date>] Wildcard characters can be used as part of the specified application or sub-application names, as follows: ■ * represents any number of characters. ■ ? represents any single character.
LISTAJFTAB	Lists jobs in the Active Jobs database that were ordered from the specified SMART Folder. ctmpsm -LISTAJFTAB <*folderName*> Wildcards can used as part of the specified folder name. ■ * represents any number of characters. ■ ? represents any single character.
SCHEDTAB	Lists SMART Folders and jobs defined in the Control-M/Server database, and allows you to add or delete SMART Folders. ctmpsm -SCHEDTAB {-LISTFOLDER <*folderName*>|-UPDATE <*rowNumber*> <*udailyName*>| -ADD <*folderName*> <*udailyName*>|-DUDAILY <*rowNumber*>| -REMOVE <*folderName*>|-LISTJOBS <*rowNumber*>}

Mode	Description	
	-LISTFOLDER	Lists all instances of SMART Folders that match the specified name or mask. For example, if a SMART Folder is ordered by two different user dailies, that folder will appear twice in the output of this option. Wildcards can used as part of the folder name for this option. ▪ * represents any number of characters. ▪ ? represents any single character.
	-UPDATE	Updates the Order method name for a specific SMART Folder.
	-ADD	Adds a Order method to an existing folder. The Folder name and Order method name must be specified when this option is used.
	-DUDAILY	Removes an instance of a SMART Folder from the Control-M/Server database. If the specified instance is the only instance of the folder (that is, that folder is ordered by only one order method), the SMART Folder and all its associated jobs are deleted. If the specified instance is not the only instance of the folder, then only the specified instance is removed from the Control-M/Server database.
	-REMOVE	Deletes a specific SMART Folder and its associated jobs.
	-LISTJOBS	Lists jobs in a SMART Folder.
LISTOUTPUT	List the OUTPUTs for an order ID. The list can be restricted by runcount number. ctmpsm -LISTOUTPUT <*orderID*> [OUTPUTNUMBER {<*number*>|ALL}] These parameters are described below. To define a viewer to which the display of the OUTPUT of a job is redirected, specify the CTMPSM_VIEWER parameter in the **~<controlm_owner>/ctm_server/data/config.dat** file. For more information about the CTMPSM_VIEWER parameter, see System configuration.	
	order_ID	Identifies the job whose OUTPUTs are listed.
	number	Restricts the list to the OUTPUT whose runcount is specified. If ALL is specified, the output will contain only a list of all OUTPUTs related to the specified order ID.

Mode	Description
LISTDETAILS	Lists the details of the job associated with the specified order ID. LISTDETAILS <*orderID*>
LISTFULLDETAILS	Lists the parameters of a specified job in the Active Jobs database. In addition to the data provided by LISTDETAILS (above), LISTFULLDETAILS provides data about In conditions, Out conditions, and Variable values. (LISTFULLDETAILS was added for use with the "Zoom and Save" option in WebAccess.) ctmpsm -LISTFULLDETAILS <*orderID*>
UPDATEAJF	Performs a specified command or updates conditions for a job in the Active Jobs database that is associated with a specified order ID. ctmpsm -UPDATEAJF <*orderID*> <*command*> command is one of the following: ▪ HOLD _ Set the status of a job to HELD. ▪ FREE _ Free a previously held job. ▪ DELETE _ Mark a job for deletion. Jobs with Executing or Submitted status cannot be deleted. ▪ UNDELETE _ Undelete a job marked for deletion. ▪ RERUN _ Rerun a job. ▪ CONFIRM _ Confirm submission of a job. ▪ SET TO OK _ Set the status a job to be OK. SET TO can only be applied to inactive jobs (that is, jobs that are not running). ▪ STATISTICS – Display a job's statistics. ▪ CONDADDIN <*cond*> <*date*> <AND|OR> _ Add the specified IN condition with the specified date reference. You can include one or more additional IN conditions by using the AND or OR conjunctional parameter. ▪ CONDADDOUT <*cond*> <*date*> <+|-> _ Add the specified OUT condition with the specified date reference. Use + to indicate that the condition must be present. Use - to indicate that the condition must not be present. ▪ CONDDELIN <*cond*> _ Delete the specified IN condition. ▪ CONDDELOUT <*cond*> _ Delete the specified OUT condition. Conditions specified using this mode apply only to the specified instance of the job in the Active Jobs database. Subsequent orders of that job are not affected.

Mode	Description
UPDATESUB-APPLICATION	Applies a specified command to jobs in the specified sub-application that are associated with the specified application. ctmpsm -UPDATESUB_APPLICATION <*application*> <*sub_application*> <*command*> <*command*> is one of the following: ▪ HOLD _ Set the status of a job to HELD. ▪ FREE _ Free previously held jobs. ▪ DELETE _ Mark the jobs for deletion. ▪ UNDELETE _ Undelete the jobs marked for deletion. ▪ CONFIRM _ Confirm submission of the jobs.
UPDATEFOLDER	Applies a specified command to jobs in the Active Jobs database that were ordered from the specified folder. ctmpsm -UPDATEFOLDER <*folder*> <*command*> <*command*> is one of the following: ▪ HOLD _ Set the status of a job to HELD. ▪ FREE _ Free previously held jobs. ▪ DELETE _ Mark the jobs for deletion. ▪ UNDELETE _ Undelete the jobs marked for deletion. ▪ RERUN _ Rerun the jobs. ▪ CONFIRM _ Confirm submission of the jobs.

Mode	Description

XML	Lists jobs in the Active Jobs database in XML format. The list can be filtered by order date, time, application, and member name. The list can be sorted by order ID or job name.
	ctmpsm -XML [{ODAT|TIME|APPLICATION|MEMNAME|ALL|ALLFIELDS}] [-SORT <ORDERID|JOBNAME>]
	To list jobs in the Active Jobs database in XML format, specify **ctmpsm -XML** plus at least one of the following fields:
	▪ ODAT – Order date.
	▪ TIME _ Time execution started and ended.
	▪ APPLICATION _ Application to which the job belongs.
	▪ MEMNAME _ Member name for the job.
	▪ ALL – Includes ODAT and TIME fields.
	▪ ALLFIELDS – Includes ODAT, MEMNAME, and APPLICATION fields.
	In addition, you can specify the following:
	–SORT indicates the order in which the jobs should be listed. Valid values: ORDERID and JOBNAME.
	To filter the list of jobs in the Active Jobs database according to the member name of the job, specify the following:
	`ctmpsm -XML MEMNAME`
	To sort the list in Example 1 above according to job name, specify the following:
	`ctmpsm -XML MEMNAME -SORT JOBNAME`

To display the most recent OUTPUT of the job whose order ID is 1234, specify the following command:

```
ctmpsm -listoutput 1234
```

To display the second OUTPUT of the job whose order ID is 1234, specify the following command:

```
ctmpsm -listoutput 1234 -outputnumber 2
```

ctmsweep

The ctmsweep utility deletes job definitions from the Control-M/Server database that become obsolete due to the Start date (Active From Date) and End date (Active To Date) parameters in the job definitions. To run the ctmsweep utility, see Running the ctmsweep utility (on page 236).

In a typical production situation, you can use ctmsweep as follows:

- Run the ctmsweep utility with the –Test parameter to generate the **sweep_obsolete.txt** file, which is essentially a report listing the obsolete jobs and folders.
- Check the **sweep_obsolete.txt** file and decide if you want to delete the jobs and folders that are displayed in the report.
- Activate the ctmsweep utility without the –Test parameter to delete the obsolete jobs and folders from the Control-M/Server database.

If the utility failed or partially succeeded, check the **U_CTMSWEEP.<PID>.log** file for the reason for the failure. Perform the necessary corrections. Activate the ctmsweep utility again to delete the obsolete folders and jobs that were not deleted successfully in the previous run.

Avoid updating job or folder definitions when the ctmsweep utility is running.

The ctmsweep utility can be invoked using the -input_file parameter:

```
ctmsweep -input_file <fullPathFileName>
```

The referenced file contains all the required parameters. Most of the parameters are described in Control-M Parameters and ctmsweep parameters *(on page* 237*).*

Running the ctmsweep utility

This procedure describes how to run the ctmsweep utility, which enables you to delete job definitions from the Control-M/Server database that become obsolete due to the Start date (Active From Date) and End date (Active To Date) parameters in the job definitions

➢ To run the ctmsweep utility:

- Type the following command:

```
ctmsweep [-Date <yyyymmddDate>]
                     [-Test ]
                     [-H | -Help]
                     [-DEBUG <debugLevel 0-5>]
                     [-MODULE <moduleNumber 0-3>]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmsweep parameters, see ctmsweep parameters (on page 237).

ctmsweep parameters

The following table lists the ctmsweep utility parameters.

Parameter	Description
-Date	Sets the date selection criteria for obsolete jobs The format of the -Date parameter is <*yyyymmddDate*>. The default is two days before the current date.
-Test	Causes the ctmsweep utility to scan all job definitions and generate the **sweep_obsolete.txt** file, which consists of a report of the current obsolete jobs and folders, without actually deleting the jobs
-H | -Help	Displays the usage
-DEBUG	Activates a debug trace at the specified level Valid debug trace levels are 0–5. The default is **0**. Performance is slower when Control-M/Server is operating in debug mode. BMC recommends that you activate debug mode only when requested by Customer Support.
-MODULE	Indicates which components are to be traced for diagnostic purposes Valid values are as follows: ▪ 0 – all components ▪ 1 – common functionality flow (default) ▪ 2 – event manager ▪ 3 – database layer

Utility reports

The following table lists the files created by the ctmsweep utility for each obsolete folder ot job that is deleted:

File name	Location	Description
sweep_obsolete.txt	<*Control-M ServerHomeDirectory*>	report of all jobs and folders that are candidates for deletion according to the date criteria
U_CTMSWEEP.<PID>.log	<*Control-M ServerHomeDirectory*>**/ctm_server/proclog**	execution log

Report formats

The report for obsolete jobs has the following format:

```
Obsolete folders/jobs [on 03/18/2010 09:56:32] for date: 20100316

Command    Folder/Job No Mem Name           Name
Folder Name
```

The report is relevant for jobs, Sub-folders and SMART Folders.

Only the most recent version of the **sweep_obsolete.txt** report is saved.

ctmsweep return codes

The following table lists the ctmsweep return codes:

Return code	Description
0	Success (All of the obsolete jobs were deleted.)
1	Failure Run the ctmsweep utility again.
2	Partial success (Some obsolete jobs were not deleted.) Run ctmsweep again to delete the remaining jobs or folders.

The CTM_MAX_OBSOLETE_JOBS parameter, in the Control-M/Server **config.dat** file, determines the maximum number of obsolete jobs that the ctmsweep utility processes when running. Valid values are from 1 to 500,000 jobs. The default is 100,000 jobs.

ctmsweep obsolete criteria

The following lists the criteria that ctmsweep uses to determine whether a job or folder is obsolete.

If the DateUntil parameter is specified, the following elements are considered as obsolete:

A regular job, which is not part of a SMART Folder, is considered as obsolete if one of the following conditions is true:

- The job has no Rule-Based Calendars and its own DateUntil/DateFrom satisfies the following criteria:

  ```
  DateFrom <= DateUntil and DateUntil < obsolete date
  ```

- The job has Rule-Based Calendars and is considered obsolete according to ctmsweep delete a job criteria (on page 240).
- A Rule-Based Calendar is considered obsolete if the following criteria is true:

  ```
  DateFrom <= DateUntil and DateUntil < obsolete date
  ```

- A SMART Folder is considered obsolete if all of its Rule-Based Calendars are obsolete (the relationship between the Rule-Based Calendars is always OR).
- A folder is considered obsolete if all of the jobs in the folder are obsolete.
- A job that is part of a SMART Folder is defined as obsolete if one of the following conditions is true:
- The SMART Folder is obsolete.
- The job satisfies the obsolete criteria according to its own DateUntil/DateFrom and has no Rule-Based Calendars.
- The job has Rule-Based Calendars and is considered obsolete according to ctmsweep delete a job criteria (on page 240).

ctmsweep delete a job criteria

The following table lists the criteria for deleting a job by the ctmsweep utility:

Relationship	Job (DateUntil/ DateFrom)	Rule-Based Calendar (DateUntil/ DateFrom)	Result
AND	obsolete	obsolete	Delete the job.
AND	obsolete	active	Delete the job.
AND	active	obsolete	Delete the job.
AND	active	active	The job is active.
OR	obsolete	obsolete	Delete the job.
OR	obsolete	active	The job is active.
OR	active	obsolete	The job is active.
OR	active	active	The job is active.
AND	not defined	obsolete	Delete the job.
AND	not defined	active	The job is active.
OR	not defined	obsolete	The job is active.
OR	not defined	active	The job is active.

To delete obsolete folders or jobs, you need both **Delete** and **Update** permissions for the specific folder.If permission is denied for a specific folder, the ctmsweep utility ignores the folder, continues processing other folders in the database, and (if no other errors are encountered) exits with a success return value.

ctmwhy

The ctmwhy utility displays a report stating why a SMART Folder, Sub-folder, or job waiting in the Active Jobs database is not being submitted for execution. This utility is equivalent to the **Why** option available when right-clicking a SMART Folder, Sub-folder, or job in the Tree View pane of the Control-M/EM window. To run the ctmwhy utility, see Running the ctmwhy utility (on page 241).

The ctmwhy utility can be invoked using the -input_file parameter:

ctmwhy -input_file <*fullPathFileName***>**

The referenced file contains all the required parameters. Most of the parameters are described in Control-M Parameters*.*

Running the ctmwhy utility

This procedure describes how to run the ctmwhy utility, which enables you to display a report stating why a SMART Folder, Sub-folder, or job waiting in the Active Jobs database is not being submitted for execution.

To run the ctmwhy utility:

- Type the following command:

```
ctmwhy <orderID>
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmsweep parameters, see ctmwhy example (on page 241).

If the host is upgrading, and the job is in state wait host, the following message will appear:

No agent hosts availabe for job's host-group.

The variable *<orderID>* is the Order ID of a job waiting in the Active Jobs database (as displayed in the Job Details window of Control-M/EM).

The Order ID as displayed in the Job Details window is a base 36 number. If you wish to specify the Order ID here as a base 10 number, prefix the number with an asterisk, and enclose the result in quotation marks (for example, " *1234").

ctmwhy example

Specify the following command to determine why the job with Order ID A4X is not being submitted for execution:

```
ctmwhy A4X
```

A typical response might be QR: 'TAPE4' : needed 2. None reserved. This response indicates that the job is not being submitted because it requires two of the Quantitative resource TAPE4 and none is available.

Specify the following command to determine why the job with Order ID 11 is not being submitted for execution. The Order ID in this example is expressed as a base 10 number:

```
ctmwhy "*37"
```

Control-M/Agent utilities

This table lists the Control-M/Agent utilities that are used for definition, ordering, and monitoring.

Utility Type	Description
_exit (on page 242)	(*Windows only*) The _exit utility is similar to the **exit** built-in shell function in UNIX.
_sleep (on page 242)	(*Windows only*) The _sleep utility is similar to the **sleep** built-in shell function in UNIX.

_exit

(*Windows only*) The _exit utility is similar to the **exit** built-in shell function in UNIX.

This utility is located in the **<Control-M/Agent>\EXE** directory path that was created during the installation procedure.

To run the _exit utility, see Running the _exit utility (on page 242).

Running the _exit utility

This procedure describes how to run the _exit utility.

To run the _exit utility:

1. Type the following command:

   ```
   _exit [<exitCode>]
   ```

   The <*exitCode*> variable is any whole integer number *n*   Default: **0**

   The program exits with %errorlevel% = <*exitCode*>.

2. Specify _exit 0 in a script to cause the job to end with %errorlevel% 0.

   ```
       ctmcreate -what command -cmdline "_exit 0"
   ```

3. Specify _exit 1 in a script to cause the job to end with %errorlevel% 1.

   ```
       ctmcreate -what command -cmdline "_exit 1"
   ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

_sleep

(*Windows only*) The _sleep utility is similar to the **sleep** built-in shell function in UNIX.

This utility is located in the **<Control-M/Agent>\EXE** directory path that was created during the installation procedure.

To run the _sleep utility, see Running the _sleep utility (on page 242).

Running the _sleep utility

This procedure describes how to run the _sleep utility.

To run the _sleep utility:

1. Type the following command:

   ```
   "... _sleep" <seconds>
   ```

   The <*seconds*> variable is any whole integer number *n*.

2. Run the following command to suspend execution of the script for 5 seconds.

   ```
       ctmcreate -what command -cmdline "_sleep 5"
   ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Chapter 3

Folders and Calendars

The folders and calendars utilities are used to manage:

- Folder definitions
- SMART Folder definitions
- Calendar definitions

You can use the Folder Manager, Job Properties team, and Folder Properties team in Control-M Workload Automation and Control-M/Enterprise Manager (Control-M/EM) for the same tasks. However, if you perform these tasks by including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utility		Description
ctmcalc_date (on page 286)		Calculate the date a job can be ordered, after adding or deducting a specified number of days.
ctmdeffolder (on page 289)		Create a definition for a new SMART Folder.
ctmdefsubfolder (on page 292)		Create a definition for a new Sub-folder.
ctmrpln (on page 295)		Create a report that lists jobs in a folder and when they are scheduled to run.
emdef utility for folders and calendars (on page 245) (The emdef utility parameters are listed in the right column.)	copydefcal (on page 245)	Create a calendar in the Control-M/EM database identical to an existing calendar.
	defcal (on page 249)	Import a calendar into the Control-M/EM database.
	deffolder (on page 257)	Import folders and SMART Folders into the Control-M/EM database.
	exportdefcal (on page 279)	Export calendars in the Control-M/EM database to an file for use as input to other utilities.
	exportdeffolder (on page 298)	Export folders from the Control-M/EM database to a file.

Utility		Description
	updatedef (on page 122)	Update job processing, folders, SMART Folders and Sub-folders.

emdef utility for folders and calendars

The emdef is a command line utility used to make various modifications to folder and calendar definitions in the Control-M/EM database. The emdef uses the following parameters:

Parameter	Description
copydefcal (on page 245)	Copies jobs from one folder to another folder
defcal (on page 249)	Imports jobs into an existing folder
deffolder (on page 257)	Deletes job definitions
exportdefcal (on page 279)	Creates a copy of job definitions
exportdeffolder (on page 298)	Exports job definitions
updatedef (on page 122)	Modifies jobs, folders, and group attributes

The emdef utility manages Rule-Base Calendars similar to other calendar types; the Rule-Base Calendar parameters are specified in the arguments file.

For emdef parameters related to jobs see emdef utility for jobs (on page 40)

copydefcal

The copydefcal utility creates a new calendar definition in the Control-M/EM database identical to an existing calendar definition. Calendars can be copied and saved under different names in the same data center. Calendars in one data center can be copied to a different data center and saved under the same or different names. To run the copydefcal utility, see Running the copydefcal utility (on page 246).

Multiple calendars can be selected and copied using the * wildcard character. For an explanation of how wildcards function in XML-based utilities, see Abbreviations and conventions (on page 10).

When copydefcal is invoked, a file of arguments that you created is processed. This arguments file contains statements that specify an existing calendar or group of calendars. The specified calendars are exported to an output file. copydefcal reads arguments directly from a plain text arguments file (in XML format) instead of reading them from the command line. For more information, see copydefcal arguments file (on page 247).

Running the copydefcal utility

This procedure describes how to run the copydefcal utility, which enables you to create a new calendar definition in the Control-M/EM database identical to an existing calendar definition.

➢ To run the utility:

1. Do one of the following:

   a. Log on as a Control-M for Databases user.

   a. Open a command prompt window (Windows users, only). You do not need to be in the Control-M for Databases directory.

2. Enter either of the following commands:

   -
     ```
     emdef copydefcal [-u <userName> [-p <password>] | -pf <passwordFile>]
     -s <guiServerName> -arg <argFileName>
     ```

   -
     ```
     emdef copydefcal [-USERNAME <userName> [-PASSWORD <password>] |
     -PASSWORD_FILE <passwordFile>] -HOST <guiServerName> -ARG_FILE
     <argFileName>
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the copydefcal parameters, see copydefcal parameters (on page 246).

The copydefcal arguments file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

copydefcal parameters

The following table lists the copydefcal utility parameters:

Parameter	Description
<user>	Control-M/EM user name.
<password>	The Control-M/EM user password.

<passwordFile>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

<guiServerName >	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<argFileName>	Path and name of the arguments file containing the copydefcal specifications. Instructions for preparing this file are in XML file preparation (on page 616).

copydefcal arguments file

Arguments are used as selection criteria to determine which calendars are exported. Arguments are written to the copydefcal argument file. The arguments files created for use with the copydefcal utility are written in XML format and saved in a text file. The format in which this file must be written is described on the following pages.

When this file is invoked, calendar definitions are exported from the Control-M/EM database.

The following rules apply to the copydefcal arguments file:

- More than one calendar can be specified in an arguments file.
- The arguments file is case-sensitive.
- Only one COPYCAL parameter can be used in an arguments file.
- All parameter values (such as strings or digits) must be enclosed in quotation marks (for example, JOBNAME="Job1").

Using FROM and TO in the copydefcal input file:

- If you choose to specify a parameter, the FROM subparameter is mandatory, and the TO subparameter is optional.
- When a FROM value is specified without a TO value, it is used as a filter criterion.
- When a TO value is included, it indicates the new value with which the parameter is updated.
- Multiple values can be specified for TO and FROM by using the * wildcard character. For an explanation of how wildcards function in the XML-based utilities, see Abbreviations and conventions (on page 10).
- If any FROM value contains *, and the corresponding TO value contains *, the * in the TO value expresses the same information as the * in the FROM value.
- The TO attribute of the DATACENTER parameter must be used to import the copied calendar into a different data center if the copy has the same name as the original calendar (the TO attribute is not used with the CALENDAR parameter). Otherwise, the calendar copy overwrites the original in the same data center.

For more information on the arguments file parameters for the copydefcal utility, see copydefcal arguments file parameters (on page 248) and copydefcal arguments file examples (on page 248).

copydefcal arguments file parameters

The following table lists the copydefcal utility arguments file parameters:

Parameter	Description
The first two lines of the XML request file for this API request contain information that specifies the version of XML, the text encoding format being used, and the location of the **.dtd** file.	
COPYCAL	These tags indicate the start and end of the COPYCAL argument. Only criteria that are located between the tags are considered to be part of the argument.
DATACENTER	Control-M installation to which the calendar definition belongs. The COPYCAL element must contain only one DATACENTER parameter. String. `DATACENTER FROM="EM5NYC" TO="EM7NYC"`
	FROM.Data center in which the source calendar is located. String. Mandatory.
	TO. Data center in which a calendar can be created. String. Optional.
CALENDAR	Name of the calendar. The COPYCAL element must contain only one CALENDAR parameter. `CALENDAR FROM="Cal1" TO="Cal1_COPY"`
	FROM. Name of the calendar from which a copy is made. String. Mandatory.
	TO. Name of the calendar copy. The copy retains the name of the original calendar if this attribute is not used. String. Optional.

copydefcal arguments file examples

The following are examples of argument files used with the copydefcal utility:

Create and import a calendar

Creates a copy of the calendar named **CAL_3** in the **EM10LA** data center.

```
<COPYCAL>
<DATACENTER FROM="EM5NYC" TO="EM10LA"/>
<CALENDAR FROM="CAL_3"/>
</COPYCAL>
```

Copy multiple calendars from the same data center

All calendars in the EM5NYC data center with names beginning with the letter A are copied to the EM7NYC data center. The new calendar names are *calendarname*_COPY (for example, the copy of the Alljobs calendar is named Alljobs_COPY).

```
<COPYCAL>
<DATACENTER FROM="EM5NYC" TO="EM7NYC"/>
<CALENDAR FROM="A*" TO="A*_COPY"/>
</COPYCAL>
```

Copy a calendar and rename the copy

The calendar named CAL_NOV in the EM5NYC data center is copied. The name of the copy is CAL_NOV_REVISED.

```
<COPYCAL>
<DATACENTER FROM="EM5NYC"/>
<CALENDAR FROM="CAL_NOV" TO="CAL_NOV_REVISED"/>
</COPYCAL>
```

defcal

The defcal utility imports a calendar definition into the Control-M/EM database. defcal reads calendar definitions directly from a plain text input file (in XML format) instead of reading them from the command line. To run the defcal utility, see Running the defcal utility (on page 249).

Running the defcal utility

This procedure describes how to run the defcal utility, which enables you to import a calendar definition into the Control-M/EM database.

➢ To run the utility:

1. Do one of the following:
   - **a.** Log on as a Control-M for Databases user.
   - **a.** Open a command prompt window (*Windows*).
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter either of the following commands:
   - `emdef defcal [-USERNAME <userName> [-PASSWORD <password>] | -PASSWORD_FILE <passwordFile>] -HOST <guiServerName> -SRC_FILE <srcFileName>`
   - `emdef defcal [-u <userName> [-p <password>] | -pf <passwordFile>] -s <guiServerName> -src <srcFileName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the defcal parameters , see defcal parameters (on page 250).

The defcal input file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

defcal parameters

The following table lists the defcal utility parameters:

Parameter	Description
<*userName*>	Control-M/EM user name.
<*password*>	The Control-M/EM user password.

<*passwordFile*>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

<*guiServerName*>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<*srcFileName*>	The path and name of the XML file containing the defcal specifications.

defcal input file

The calendars that you create for use with the defcal utility are written in XML format and saved in a text file. When this file is invoked, its contents are passed to the Control-M/EM database.

Instructions for creating an XML format input file are in XML file preparation (on page 616).

The following rules apply to the defcal utility input file:

- More than one calendar can be specified in a defcal file.
- The XML file is case-sensitive.
- The definition for a single calendar can cover a period of one or more years.
- All parameter values (such as strings or digits) must be enclosed in quotation marks (for example, JOBNAME="Job1").

For more information on the input file parameters for the defcal utility, see defcal input file parameters (on page 251) and defcal input file examples (on page 255).

defcal input file parameters

The following table lists the defcal utility input file parameters:

Parameter	Description	
The first two lines of the XML request file for this API request contain information that specifies the version of XML, the text encoding format being used, and the location of the .dtd file.		
DEFCAL	Indicates to Control-M for Databases that the defcal utility is being initiated. Calendar definitions are placed between the opening and closing DEFCAL tags. One or more calendars can be specified.	

CALENDAR	Indicates the opening and closing tags of a single calendar definition. The parameters of the job are listed between the tags.
	`<CALENDAR DATACENTER="EM5A" NAME="AcctJob1"`
	`        TYPE="Relative"></CALENDAR>`	

	DATACENTER	Name of the Control-M installation to which the calendar definition belongs. String. Mandatory.
	NAME	Name of the calendar. String Mandatory.
	TYPE	Calendar type. Mandatory. Valid values: ■ **Regular** ■ **Periodic** ■ **Relative**

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="2"><em>YEAR</em></td><td colspan="2">Year-specific definitions in the calendar. Mandatory.
<pre>
YEAR NAME="2008"
        DAYS="YYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYYYYY
        YYY
        YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
        YY
        YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNNYYYYYYYY
        YY
        YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
        YY
        YYYNNYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
        YY
        YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
        YY
        YYYYYYYYYYYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYY
        YY
        YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNN
        YY YYYYYYYYYYYYYYYYYYY"
DESCRIPTION="This is the Accounting Jobs calendar for
        2008."
</pre></td></tr>
<tr><td>NAME</td><td>Year for which the calendar definition applies. String. Mandatory.
The year during which the calendar is active. Only one year can be entered for this attribute, but more than one YEAR parameters can be included in a calendar definition.
The value of this attribute is expressed as YYYY (for example, 2008).</td></tr>
</table>

Parameter	Description	
	DAYS	Days on which the job is ordered. String. Mandatory.Valid values: For a Relative calendar: ▪ Y ▪ N ▪ + ▪ - For a Regular calendar: ▪ Y ▪ N For a Periodic calendar: any character other than Y, N, +, or -. Each Y and N represents a single day of the year. The value of the DAYS parameter is 365 characters long (366 for a leap year). The first letter of the DAYS value is January first. The last letter is December 31.
	DESCRIPTION	Text description of the calendar. String. Optional. For Regular and periodic calendars, only.
RULE_BASED _CALENDAR	Indicates the opening and closing tags of a single Rule-Based Calendar definition. The parameters of the job are listed between the tags. `<RULE_BASED_CALENDAR DATACENTER="EM5A" NAME="AcctJob1"></RULE_BASED_CALENDAR>`	
	DATACENTER	Name of the Control-M installation to which the calendar definition belongs. String. Mandatory.
	NAME	Name of the calendar. String Mandatory.
	DAYS	Days of the month on which the job is ordered. String. Mandatory.
	DAYS_AND_OR	Indicates the relationship between specified Days values and Weekdays values. Optional. Valid values: ▪ OR ▪ AND Default: OR

Parameter	Description	
	WEEKDAYS	Days of the week on which to order the job. String. Optional. Valid values: Weekdays - 0 - 6 - ALL - blank
	DATE	Specific dates on which to order the job. String. MMDD format. Optional.
	DAYSCAL	Name of a user-defined calendar used to specify a set of days. String. Optional.
	WEEKSCAL	Name of a calendar to be used to validate specified weekdays on which to order the job. String. Optional.
	CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the job. String.
	SHIFT	Sub-parameter of CONFCAL. Describes how to shift the scheduling date of the job. Valid values: - IGNOREJOB - PREVDAY - NEXTDAY - NOCONFCAL
	SHIFTNUM	Sub-parameter of CONFCAL. Number of days to shift the scheduling date of the job. Default: +00
	MAXWAIT	Number of extra days (beyond the original scheduling date) that the job is allowed to remain in the Active Jobs database awaiting execution. Integer. Default: 00
	JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC	Months when the job can run. Optional. Not including a month is the same as including a month having the value 0. Valid values: - 0 (Default) - 1

Parameter	Description	
	ACTIVE_FROM	Starting date of the Rule-Based Calendar. Format: YYYYMMDD
	ACTIVE_TILL	Ending date of the Rule-Based Calendar. Format: YYYYMMDD

defcal input file examples

The following are examples of argument files used with the defcal utility:

Import a regular calendar

Regular calendar, named AcctCal3, is imported into the EM5NY data center.

```
<DEFCAL>

<CALENDAR

DATACENTER="EM5NY"

NAME="AcctCal3"

TYPE="Regular">

<YEAR

NAME="2008"

DAYS="YYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNNYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYNNYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNNY
YYYYYYYYYYYYYYYYYYYY"

DESCRIPTION="Calendar for 2008."/>

</CALENDAR>

</DEFCAL>
```

Import two calendars into different data centers

Two calendars are imported, each into a different data center, with a single defcal input file.

```
<DEFCAL>

<CALENDAR

DATACENTER="EM5NY"

NAME="AcctCal3"
```

```
TYPE="Regular">

<YEAR

NAME="2008"

DAYS="YYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNNYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYNNYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNNY
YYYYYYYYYYYYYYYYYYYY"

DESCRIPTION="Calendar for 2008."/>

<CALENDAR

DATACENTER="EM2LA"

NAME="HRCal3"

TYPE="Regular">

<YEAR

NAME="2008"

DAYS="YYYYYYYYYYYYYYYYNNNYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYNNNNNYYYYYYYYYYYYYYYYNNNYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYNNYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYNYYYYYYYYYYYYYY
YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYNNNY
YYYYYYYYYYYYYYYYYNNN"

DESCRIPTION="Calendar for 2008."/>

</CALENDAR>

</DEFCAL>
```

deffolder

The deffolder utility imports folders and SMART Folders into the Control-M/EM database. To run the deffolder utility, see Running the deffolder utility (on page 257).

When deffolder is invoked, a file of arguments that you have created is processed. This input file contains statements that specify:

- an existing folder or set of folders
- an existing SMART Folder or set of SMART Folders.

For more information, see deffolder input file (on page 259). The specified folders are imported into the Control-M/EM database.

If the folders do not exist in the Control-M/EM database, the utility creates them. If the folders do exist, a message is issued indicating that the folders already exist (unless the /o switch is specified, in which case the folders are overwritten – the /o switch is described below).

The deffolder utility reads folder and SMART Folder definitions directly from a plain text arguments file (in XML format) instead of reading them from the command line.

A single deffolder input file can contain specifications for both folders and SMART Folders.

XML is comprised of elements and attributes. Each element can contain attributes and sub-elements. In the folder that follows, elements are bolded and attributes are italicized.

Running the deffolder utility

This procedure describes how to run the deffolder utility, which enables you to import folders and SMART Folders into the Control-M/EM database.

➢ To run the utility:

1. Do one of the following:
   - **a.** Log on to a Control-M/EM account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the `<EM Home>\<instanceName>\bin` directory.
2. Enter either of the following commands:
   - `emdef deffolder [-u <userName> [-p <password>] | -pf <passwordFile>] -s <guiServerName> -src <srcFileName> [/a] [/o]`
   - `emdef deffolder [-USERNAME <userName> [-PASSWORD <password>] | -PASSWORD_FILE <passwordFile>] -HOST <guiServerName> -SRC_FILE <srcFileName> [/a] [/o]`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the deffolder parameters and switches, see deffolder parameters (on page 258) and deffolder switches (on page 258).

The deffolder input file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with the errors.

deffolder parameters

The following table lists the deffolder utility parameters:

Parameter	Description
<userName>	Control-M/EM user name.
<password>	The Control-M/EM user password.

<passwordFile>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

<guiServerName>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<srcFileName>	Path and name of the file containing the specifications for the folder that you are importing. Instructions for preparing this file are in XML file preparation (on page 616).

deffolder switches

The following table lists optional switches for the deffolder utility:

Switch	Description
/a	Accept all. The **/a** switch directs the utility to automatically reset the Created By parameter to the current Control-M/EM user when these two values do not match. If not specified, the utility skips (that is, does not process) job definitions whose Author does not match the currently logged in user. The ***/*a** switch has no effect on Administrator users and is relevant only when the AuthorSecurity system parameter is set to **2** or **3**.
/o	Overwrite. The **/o** switch directs the utility to overwrite any existing folders.
/t	Operate on a single folder at a time, to reduce process memory.
/v	Used to receive verbose messages.

deffolder input file

The *srcFileName* input file contains the definition of a folder. The file is written in XML format and saved in a text file. The format in which this file must be written is described on the following pages.

When this file is invoked, either folder definitions are imported to the Control-M/EM database, or SMART Folder definitions are imported, or both folder definitions and SMART Folder definitions are imported the Control-M/EM database. For instructions for creating input files, see XML file preparation (on page 616).

The following rules apply to the deffolder input file:

- Only one SMART Folder can be included in a folder. However, multiple folders can be included in a single input file. Each of these folders can contain one SMART Folder.
- Multiple SMART Folders can be included in a file if the file only contains SMART Folders.
- More than one job can be specified in either type of folder.

For more information on the input file parameters for the deffolder utility, see the following:

- deffolder input file parameters for folders (on page 260)
- deffolder input file parameters for SMART folders (on page 263)
- deffolder input file examples (on page 274)

deffolder input file parameters for folders

The following table lists the deffolder input file parameters for folders:

Parameter	Description
The first two lines of the XML request file for this API request contain information that specifies the version of XML, the text encoding format being used, and the location of the **.xsd** file.	
DEFFOLDER	Indicates to Control-M for Databases the beginning and end of the deffolder utility. Folder definitions are placed between the opening and closing DEFFOLDER tags. One or more folders can be specified. Each individual folder is enclosed by the <FOLDER><FOLDER> tags. If you are using Control-M version 6.4.00 or earlier the parameter SCHED_FOLDER will be used in place of FOLDER.
FOLDER	Indicate the opening and closing tags of a single folder definition. The parameters of the folder are listed between the tags. In the case of the folder, the folder parameters consist of parameters that describe the folder directly and a list of the jobs that are included in the folder. In turn, each of the jobs that is listed includes all of its own descriptive parameters. If you are using Control-M version 6.4.00 or earlier the parameter SCHED_FOLDER will be used in place of FOLDER.
FOLDER_NAME	Name of the folder to which the job belongs. String. Mandatory. The following folder parameters must be specified for each folder: ▪ DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN (z/OS only)
FOLDER_DSN	Name of the library that contains the folder. [For z/OS jobs, only.] String. Optional. The following folder parameters must be specified for each folder: ▪ DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN (z/OS only)
DATACENTER	Name of the Control-M installation to which the folder belongs. String. Mandatory. The following folder parameters must be specified for each folder: ▪ DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN (z/OS only)

Parameter	Description
FOLDER_ORDER_ METHOD	The Newday or User-daily name. Optional.
USED_BY	For internal use. Do not include this parameter in your **deffolder** input file.
USED_BY_CODE	For internal use. Do not include this parameter in your **deffolder** input file.
MODIFIED	For internal use. Do not include this parameter in your **deffolder** input file.
LAST_UPLOAD	Date of the last folder upload. String. Optional.
CHECKSUM	For internal use. Do not include this parameter in your deffolder input file.
FOLDER_ID	For internal use. Do not include this parameter in your deffolder input file.
REAL_FOLDERID	For internal use. Do not include this parameter in your deffolder input file.
JOB	Opening and closing tags of a single job definition. Parameters of the job are listed between the tags. For a complete listing of defjob parameters, see defjob (on page 41).
CYCLIC_TYPE	Determines the type of cyclic job: ▪ Interval ▪ IntervalSequence ▪ SpecificTimes Relevant for Control-M version 6.4.01 or later.
CYCLIC_ TOLERANCE	Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes). Relevant for Control-M version 6.4.01 or later.
CYCLIC_INTERVAL_ SEQUENCE	A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas. Relevant for Control-M version 6.4.01 or later.
CYCLIC_TIMES_ SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.
ENFORCE_VALIDATION	Determines if validation is either an error or a warning. Valid Values: Y or N Relevant for Control-M Workload Change Manager only.

Parameter	Description
SITE_STANDARD_NAME	Defines the name of the site standard that is applied to the folder and all of its entities. For more information see Site standards management Relevant for Control-M Workload Change Manager only.
BUSINESS_PARAMETER_NAME	Defines the name of the Business Parameter name that is applied to the folder and all of its entities. For more information, see: Site standards management Relevant for Control-M Workload Change Manager only.
VALUE	Defines the value of a customer defined business field. String. Relevant for Control-M Workload Change Manager only.

deffolder input file parameters for SMART folders

The following table lists the deffolder input file parameters for SMART folders:

Parameter	Description
The first two lines of the XML request file for this API request contain information that specifies the version of XML, the text encoding format being used, and the location of the **.xsd** file.	
DEFFOLDER	Indicates the beginning and end of the deffolder utility. Folder definitions are placed between the opening and closing DEFFOLDER tags. One or more jobs can be specified. Each individual job is enclosed by the FOLDER</FOLDER tags
SMART_FOLDER	Opening and closing tags of a SMART Folder definition. If you are using Control-M version 6.4.00 or earlier the parameter SCHED_GROUP will be used in place of SMART_FOLDER.
FOLDER_NAME	Name of the SMART Folder to which the job belongs. String. Mandatory. ▪ The following SMART Folder parameters must be specified for each folder: DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN (z/OS only)
SUB_APPLICATION	Name of the group to which the jobs in the SMART Folder are assigned. String. Mandatory.
DATACENTER	Name of the Control-M installation to which the SMART Folder belongs. String. Mandatory. The following folder parameters must be specified for each folder: ▪ DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN (z/OS only)
FOLDER_DSN	Library for the folder. String. Optional. The following folder parameters must be specified for each folder: ▪ DATACENTER ▪ FOLDER_NAME ▪ FOLDER_DSN (z/OS only)
FOLDER_ORDER_ METHOD	String. Optional.
USED_BY	For internal use. Do not include this parameter in your deffolder input file.

Parameter	Description
USED_BY_CODE	For internal use. Do not include this parameter in your deffolder input file.
MODIFIED	For internal use. Do not include this parameter in your deffolder input file.
LAST_UPLOAD	Date of the last folder upload. String. Optional.
CHECKSUM	For internal use. Do not include this parameter in your deffolder input file.
FOLDER_ID	For internal use. Do not include this parameter in your deffolder input file.
REAL_FOLDERID	For internal use. Do not include this parameter in your deffolder input file.
JOBNAME	Name of the job processing definition.String. Optional.
MEMNAME	Name of the file that contains the job script. String. Optional.
APPLICATION	Name of the application to which the SMART Folder belongs. Used as a descriptive name for related groups of SMART Folders. String. Mandatory.
RUN_AS	Owner (user ID) associated with the SMART Folder. This parameter is used by the Control-M/Server security mechanism. String. Optional.
ADJUST_COND	Indicates whether to ignore prerequisite conditions normally set by predecessor jobs if the relevant predecessor jobs are not scheduled. This parameter is relevant only for jobs in a SMART Folder. String. Optional.
CONFIRM	Indicates that the SMART Folder must be manually confirmed by the Control-M/EM user before it runs. Valid values: ▪ 0 (No confirmation. Default.) ▪ 1 (Requires confirmation.)
PRIORITY	Indicates Control-M SMART Folder priority. String. Optional.
TIMEFROM	Indicates the earliest time for submitting the SMART Folder. String. Optional.
TIMETO	Indicates the latest time for submitting the SMART Folder. String. Optional.
DUE_OUT	Time that the jobs in the SMART Folder are expected to finish. String. Optional.
DOCMEM	Name of the file containing SMART Folder documentation. String. Optional.
DOCLIB	Name of the DOCMEM library. String. Optional.
DESCRIPTION	Brief text description of the SMART Folder. String. Optional.

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td><em>CREATED BY</em></td><td colspan="2">Control-M/EM user who defined the SMART Folder. String. Mandatory.
This argument is used by the Control-M security mechanism and, under certain circumstances, cannot be modified. For more information, see the Security chapter and the description of the <strong>AuthorSecurity</strong> system parameter in GUI Server parameters.</td></tr>
<tr><td><em>CREATION_USER</em></td><td colspan="2">Name of the user who created the SMART Folder. String. Optional.</td></tr>
<tr><td><em>CREATION_DATE</em></td><td colspan="2">Date on which the SMART Folder was created. String. Optional.</td></tr>
<tr><td><em>CREATION_TIME</em></td><td colspan="2">Time at which the SMART Folder was created. String. Optional.</td></tr>
<tr><td><em>CHANGE_USERID</em></td><td colspan="2">Name of the user who last modified the SMART Folder. String. Optional.</td></tr>
<tr><td><em>CHANGE_DATE</em></td><td colspan="2">Date on which the SMART Folder was last modified. String. Optional.</td></tr>
<tr><td><em>CHANGE_TIME</em></td><td colspan="2">Time at which the SMART Folder was last modified. String. Optional.</td></tr>
<tr><td><em>MULTY_AGENT</em></td><td colspan="2">If set to Y, job submission details are broadcasted to all agents within an Application Group. The agent with available resources runs the jobs in the SMART Folder. Optional.
Valid values:
▪ Y - run as a multi-agent job
▪ N - do not run as a multi-agent job. Default.</td></tr>
<tr><td><em>ACTIVE_FROM</em></td><td colspan="2">Indicates the start of a period of time during which the job or SMART Folder can be ordered. [For z/OS jobs and SMART Folders, only.] Optional. Date Format: YYYYMMDD</td></tr>
<tr><td><em>ACTIVE_TILL</em></td><td colspan="2">Indicates the end of a period of time during which the job or SMART Folder can be ordered. [For z/OS jobs and SMART Folders, only.] Optional. Date Format: YYYYMMDD</td></tr>
<tr><td rowspan="2">RULE_BASED_CALENDAR</td><td colspan="2">Collection of scheduling criteria organized unit with a unique name. Mandatory.
<pre>
RULE_BASED_CALENDAR
        RULE_BASED_CALENDAR_NAME="RULE_BASED_CALENDAR1"
        DAYS="1,8,15,23" DAYS_AND_OR="AND"
        WEEKDAYS="wcal_3" DATE="18" DAYSCAL=""
        CONFCAL="cal_4" RETRO="1" SHIFT="PREVDAY"
        SHIFTNUM="5" MAXWAIT="5" MAXRUNS="2" JAN="1"
</pre></td></tr>
<tr><td><em>RULE_BASED_CALENDAR_NAME</em></td><td>Unique name of the Rule_Based_Calendar. String. Mandatory.</td></tr>
</table>

Parameter	Description	
	DAYS	Days of the month on which to order the jobs in the SMART Folder. String. Optional.
	DAYS_AND_OR	Indicates the relationship between specified Days values and Weekdays values. Optional. Valid values: ▪ AND ▪ OR
	WEEKDAYS	Days of the week on which to order the jobs in the SMART Folder. String. Optional.
	DATE	Specific dates on which to order the jobs in the SMART Folder. String. mmdd format. String. Optional.
	DAYSCAL	Name of a user-defined calendar used to specify a set of days. String. Optional.
	CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the jobs in the SMART Folder. String. Optional.
	RETRO	Indicates whether the jobs in the SMART Folder is scheduled for possible execution after their original scheduling date (odate) has passed. Optional. Applies only to Rule_Based_Calendars. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)

Parameter	Description	
	SHIFT	Describes how to shift the scheduling date of the jobs in the SMART Folder. Optional. Valid values: ■ IGNOREJOB ■ PREVDAY ■ NEXTDAY ■ NOCONFCAL
	SHIFTNUM	Number of days to shift the scheduling date of the jobs in the SMART Folder. String. Optional.
RULE_BASED_CALENDAR *continued*	*MAXWAIT*	Number of extra days (beyond the original scheduling date) that the jobs in the SMART Folder are allowed to remain in the Active Jobs database awaiting execution. The value of MAXWAIT in the Rule_Based_Calendar is the value of the MAXWAIT for the jobs that use this Rule_Based_Calendar. Integer. Optional.
	MAXRUNS	Maximum number of job runs to retain the SYSDATA archive data set for jobs that ended NOTOK. Subparameter of AUTOARCH. String. Valid values: 000 – 998, or 999 to retain the archived data for all runs. Optional.
	JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC	Months when the jobs in the SMART Folder can run. Optional. Valid values: ■ 0 (Default) ■ 1

TAGs	Collection of scheduling criteria organized unit with a unique name. Mandatory. This parameter is for backward compatibility.
	`TAG TAG_NAME="tag1" DAYS="1,8,15,23" DAYS_AND_OR="AND"`
	`WEEKDAYS="wcal_3" DATE="18" DAYSCAL=""`
	`CONFCAL="cal_4" RETRO="1" SHIFT="PREVDAY"`
	`SHIFTNUM="5" MAXWAIT="5" MAXRUNS="2" JAN="1"`	

	TAG_NAME	Unique name of the tag. String. Mandatory.
	DAYS	Days of the month on which to order the jobs in the SMART Folder. String. Optional.

Parameter	Description	
	DAYS_AND_OR	Indicates the relationship between specified Days values and Weekdays values. Optional. Valid values: ▪ AND ▪ OR
	WEEKDAYS	Days of the week on which to order the jobs in the SMART Folder. String. Optional.
	DATE	Specific dates on which to order the jobs in the SMART Folder. String. mmdd format. String. Optional.
	DAYSCAL	Name of a user-defined calendar used to specify a set of days. String. Optional.
	CONFCAL	Specifies a calendar that is used to validate all specified days and dates on which to schedule the jobs in the SMART Folder. String. Optional.
	RETRO	Indicates whether the jobs in the SMART Folder is scheduled for possible execution after their original scheduling date (odate) has passed. Optional. Valid values: ▪ 0 (No. Default) ▪ 1 (Yes)

Parameter	**Description**	
	SHIFT	Describes how to shift the scheduling date of the jobs in the SMART Folder. Optional. Valid values: ▪ IGNOREJOB ▪ PREVDAY ▪ NEXTDAY ▪ NOCONFCAL
	SHIFTNUM	Number of days to shift the scheduling date of the jobs in the SMART Folder. String. Optional.
TAGs *continued*	*MAXWAIT*	Number of extra days (beyond the original scheduling date) that the jobs in the SMART Folder are allowed to remain in the Active Jobs database awaiting execution. The value of MAXWAIT in the tag is the value of the MAXWAIT for the jobs that use this tag. Integer. Optional.
	MAXRUNS	Maximum number of job runs to retain the SYSDATA archive data set for jobs that ended NOTOK. Subparameter of AUTOARCH. String. Valid values: 000 – 998, or 999 to retain the archived data for all runs. Optional.
	JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC	Months when the jobs in the SMART Folder can run. Optional. Valid values: ▪ 0 (Default) ▪ 1

Parameter	Description	

RULE_BASED_CALENDAR_NAMES	Wrapper for specifying one or more Rule-Based Calendars for the SMART Folder.
	`RULE_BASED_CALENDAR_NAMES`
	`        RULE_BASED_CALENDAR_NAME="RULE_BASED_CALENDAR_1"`	

	RULE_BASED_CALENDAR_NAME	String. Optional.
TAG_NAMES	Wrapper for specifying one or more TAGs for the SMART Folder. For backward compatibility. This parameter is for backward compatibility. `TAG_NAMES TAG_NAME="TAG_1"`	
	TAG_NAME	String. Optional.
INCOND	In condition. Optional. `INCOND NAME="Cond1" ODATE="ODAT" AND_OR="AND" OP="("`	
	NAME	Name of the In condition. String. Mandatory. 1 - 255 characters, case-sensitive.
	ODATE	Order date of the In condition. String. Mandatory. Default: ODAT
	AND_OR	Relationship between conditions. Valid values: ▪ AND (default) ▪ OR
	OP	Parentheses indicating parts of the condition that are interpreted first. String. Optional.
OUTCOND	Out condition. Optional. `OUTCOND NAME="Job1" ODATE="ODAT" SIGN="ADD"`	
	NAME	Name of the Out condition. String. Mandatory. 1 - 255 characters, case-sensitive.
	ODATE	Order date of the Out condition. String. Mandatory. Default: ODAT
	SIGN	Indicates whether to add or delete the condition. Valid values: ▪ ADD (default) ▪ DEL

Parameter	Description	
VARIABLE	Wrapper for the Variable expression. Optional. `VARIABLE EXP="%%PARM1=%%TIME"`	
	EXP	Variable expression. String. Mandatory. `%%PARM1=%%TIME`

SHOUT	Wrapper for the Shout message. Optional.
	`SHOUT WHEN="EXECTIME" DEST="workstation1" URGENCY="R"`
	`MESSAGE="Job completed OK." TIME=">10"`	

	WHEN	Condition under which the Shout message is sent. Mandatory. Valid values: ▪ OK (default) ▪ NOTOK ▪ RERUN ▪ LATESUB ▪ LATETIME ▪ EXECTIME
	DEST	Recipient of the shout message. String. Mandatory. Valid values: 1 - 16 characters, case-sensitive.
	URGENCY	Indicates the urgency of the Shout message. Mandatory. Valid values: ▪ R (regular-default) ▪ U (urgent) ▪ V (very urgent)
	MESSAGE	Text of the message. String. Mandatory. Valid values: 1 - 70 characters, spaces allowed.
	TIME	Time of the message. String. Mandatory.

Parameter	Description	
ON_SUB-APPLICATION	Folder-processing termination status, for a SMART Folder, that determines whether the accompanying DO statements are performed. `ON_GROUP CODE="OK"`	
	CODE	Indicates whether the DO statements are performed when the SMART Folder ends OK or NOTOK. Mandatory. Valid values: ▪ OK ▪ NOTOK
DO_SUB-APPLICATION	Actions to perform when the ON_GROUP condition is fulfilled. `DO_GROUP ACTION="OK"`	
	ACTION	Mandatory. Valid values: ▪ OK ▪ NOTOK
DOVARIABLE	Wrapper for the Variable expression. Optional. `DOVARIABLE EXP="%%PARM1=%%TIME"`	
	EXP	The Variable expression. String. Mandatory. `%%PARM1=%%TIME`
DOSHOUT	Shout message wrapper. Optional. `DOSHOUT DEST="Wkstn2" URGENCY="R" MESSAGE="Job5 completed OK"`	
	DEST	Recipient of the Shout message. String. Mandatory. Valid values: 1-16 characters, case-sensitive.
	URGENCY	Urgency of the Shout message. Valid values: ▪ R (regular-default) ▪ U (urgent) ▪ V (very urgent)

Parameter	Description	
	MESSAGE	Text of the Shout message. String. Mandatory. Valid values: 1 - 70 characters, spaces allowed.

DOFORCEJOB	Forces a specified job when the current SMART Folder is complete. Optional.
	`DOFORCEJOB DSN="45446" FOLDER_NAME="Folder2" NAME="Job4"`
	`ODATE="ODAT"`	

	DSN	Library for the (scheduling) folder [z/OS only]. String. Mandatory.
	FOLDER_NAME	Name of the folder to which the job belongs. String. Mandatory. Valid values: 1 - 10 characters.
	NAME	Name of the job. String. Mandatory.
	ODATE	Original scheduling date for the job. String. Default: ODAT
DOCOND	Specifies prerequisite conditions to be added or deleted. Optional. `DOCOND NAME="Cond1" ODATE="ODAT" SIGN="ADD"`	
	NAME	Condition name. String. Mandatory. Valid values: 1 - 20 characters, case-sensitive.
	ODATE	Condition date. String. Mandatory. Default: ODAT
	SIGN	Specifies whether to add or delete the condition. Valid values: ▪ ADD (default) ▪ DEL

DO MAIL	Sends mail when the SMART Folder run is complete. Optional.
	`DOMAIL URGENCY="R" DEST="emuser@emuser.com"`
	`CC_DEST="david@emuser.com" SUBJECT="OK"`
	`MESSAGE="Task completed OK."`	

	ATTACH_OUTPUT	Specifies at the job level whether the output should be sent as an email attachment.
	URGENCY	Urgency of the message. Valid values: ▪ R (regular - Default) ▪ U (Urgent)

Parameter	Description	
	DEST	Recipient of the message. String. Mandatory.
	CC_DEST	Additional recipient of the message. String. Optional.
	SUBJECT	Brief text description of the message contents. String. Optional.
	MESSAGE	Text of the message. String. Mandatory.
JOB	Indicate the opening and closing tags of a single job definition. The parameters of the job are listed between the tags. For a complete listing of defjob parameters, seedefjob. (on page 41)	

deffolder input file examples

The following are examples of argument files used with the deffolder utility:

Folder with two jobs

```
<DEFFOLDER>
<FOLDER
FOLDER_NAME="2:35"
FOLDER_DSN="KDSN"
DATACENTER="phantom">
<JOB
JOBNAME="KURT999"
MEMNAME="kurt_m"
GROUP="KGROUP"
APPLICATION="KAPP"
AUTHOR="CTMEMUSER"
TASKTYPE="Command"
MAXRERUN="1"
INTERVAL="1"
PRIORITY="1"
CRITICAL="1"
CYCLIC="1"
CONFIRM="1"
DAYS="1,2,3"
DAYSCAL="">
```

```
<SHOUT WHEN="OK" DEST="NOWHERE" MESSAGE="Job is
OK" TIME="1045"/>

</JOB>

<JOB

JOBNAME="KURT901"

MEMNAME="kurt m"

GROUP="KGROUP"

APPLICATION="KAPP"

TASKTYPE="Command"

MAXRERUN="1"

INTERVAL="1"

PRIORITY="1"

CRITICAL="1"

CYCLIC="1"

CONFIRM="1"

DAYS="1,2,3"

DAYSCAL="123">

<SHOUT WHEN="OK" DEST="NOWHERE" MESSAGE="Job is
 OK"
TIME="1045"/>

</JOB>

</FOLDER>

</DEFFOLDER>
```

SMART Folder with one job

```
<SMART_FOLDER

DATACENTER="MIG4"

FOLDER_NAME="FOLDER1"

FOLDER_DSN="FOLDERLIB"

GROUP="GRP"

JOBNAME="FOLDER1"

APPLICATION="APPL"

MEMNAME="MEMNAME"

OWNER="JACKH"

AUTHOR="CTMEMUSER"

ADJUST_COND="1"

CONFIRM="1"
```

```
PRIORITY="AB"

TIMEFROM="0900"

TIMETO="1100"

DUE_OUT="0500"

DOCMEM="DOC"

DOCLIB="A.B.C"

DESCRIPTION="desc">

<RULE_BASED_CALENDAR

RULE_BASED_CALENDAR_NAME="TEST"

DAYS="ALL"

DAYS_AND_OR="OR"

WEEKDAYS="1,2"

SHIFT="NEXTDAY"

SHIFTNUM="+24"

DAYSCAL="EYALDCAL"

WEEKSCAL="EYALWCAL"

CONFCAL="EYALCCAL"

MAXWAIT="10"

JAN="1"/>

<RULE_BASED_CALENDAR

RULE_BASED_CALENDAR_NAME="TEST2"

DAYS="1,2,3,4"

DAYS_AND_OR="AND"

WEEKDAYS="1,2"

RETRO="0"

SHIFT="PREVDAY"

SHIFTNUM="+24"

MAXWAIT="10"

FEB="1"/>

<INCOND  NAME="IN1" ODATE="$$$$"/>

<INCOND  NAME="IN2" ODATE="$$$$"/>

<INCOND  NAME="IN3" ODATE="****" AND_OR="OR
OP="O("/>

<INCOND  NAME="IN4" ODATE="STAT" AND_OR="OR"

OP=")"/>
```

```
<OUTCOND NAME="OUT1" ODATE="STAT" SIGN="DEL"/>
<OUTCOND NAME="OUT1" ODATE="STAT"/>
<VARIABLE EXP="DUMMY=ggg"/>
<ON_GROUP CODE="NOTOK">
<DOSHOUT  DEST="ShoutDest" URGENCY="U" MESSAGE="msg"/>
<DO_GROUP ACTION="OK"/>
<DO_GROUP ACTION="NOTOK"/>
<DOFORCEJOB FOLDER_NAME="A.B.C" NAME="MEMNAME" ODATE="1011"/>
<DOCONDNAME="condname" ODATE="0506"/>
<DOMAIL DEST="eyal@bmc.com" MESSAGE="hello"/>
<DOVARIABLE  EXP="A=B"/>
</ON_GROUP>
<SHOUT WHEN="OK" DEST="DestTest" URGENCY="R" MESSAGE="Message test"
TIME="1000"/>
<JOB JOBNAME="JOB_GRP" MEMNAME="JACK"
MEMLIB="JACKIB" OWNER="JACKH" APPLICATION="JACKAPP" TASKTYPE="Job"
MAXRERUN="0" INTERVAL="1" PRIORITY="1" CRITICAL="1" CYCLIC="1"
CONFIRM="1" DAYS="1,2,3" DAYSCAL="12_7" AUTHOR="JACKH">
<RULE_BASED_CALENDARS RULE_BASED_CALENDAR_NAME="TEST"/>
<OUTCOND NAME="COND1" ODATE="STAT"/>
<OUTCOND NAME="COND2" ODATE="STAT"/>
</JOB>
</SMART_FOLDER>
</DEFFOLDER>
```

deffolder input file examples: Control-M Workload Change Manager

The following example describes the Control-M Workload change manager for the defolder utility input file.

```
<DEFTABLE xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="Folder.xsd">
           <FOLDER DATACENTER="vw-tlv-em-dv131"
           VERSION="800"
           PLATFORM="UNIX"
           FOLDER_NAME="rt"
           FOLDER_ORDER_METHOD="SYSTEM" REAL_FOLDER_ID="153" TYPE="1"
           ENFORCE_VALIDATION="N" SITE_STANDARD_NAME="st_acc">
```

```
  <JOB JOBISN="1"
JOBNAME="OS_Job#1"
CREATED_BY="emuser"
RUN_AS="a"
CRITICAL="0"
TASKTYPE="Command"
CYCLIC="0"
INTERVAL="00001M"
CMDLINE="a"
CONFIRM="0"
RETRO="0"
MAXWAIT="0"
MAXRERUN="0"
AUTOARCH="1"
MAXDAYS="0"
MAXRUNS="0"
DAYS="ALL"
JAN="1"
FEB="1"
MAR="1"
APR="1"
MAY="1"
JUN="1"
JUL="1"
AUG="1"
SEP="1"
OCT="1"
NOV="1"
DEC="1"
DAYS_AND_OR="O"
SHIFT="Ignore Job"
SHIFTNUM="+00"
SYSDB="1"
IND_CYCLIC="S"
```

```
CREATION_USER="emuser"
CREATION_DATE="20130728"
CREATION_TIME="215511"
RULE_BASED_CALENDAR_RELATIONSHIP="O"
APPL_TYPE="OS"
MULTY_AGENT="N"
USE_INSTREAM_JCL="N"
VERSION_OPCODE="N"
IS_CURRENT_VERSION="Y"
VERSION_SERIAL="1"
VERSION_HOST="BMC-HSD2GV1"
CYCLIC_TOLERANCE="0"
CYCLIC_TYPE="C"
PARENT_FOLDER="rt" />
<ADDITIONAL_FOLDER_DETAILS>
  <BUSINESS_PARAMETER NAME="Dept" VALUE="Finance" />
</ADDITIONAL_FOLDER_DETAILS>
</FOLDER>
</DEFTABLE>
```

exportdefcal

The exportdefcal utility exports calendar definitions in the Control-M/EM database to an output file for use as input to other utilities. To run the exportdefcal utility, see Running the exportdeffolder utility (on page 280).

When the exportdefcal utility is invoked, an arguments file that you prepare is processed. This arguments file contains statements that specify an existing calendar or group of calendars. The calendars that you specified in the arguments file are exported to an output file. You can modify the exported calendars in the output file and re-import them into the Control-M/EM database using the defcal utility. For more information, see exportdefcal arguments file (on page 282).

Output files from export utilities (such as exportdefcal) can be used as input files with the import utilities (such as defcal).

exportdefcal reads arguments directly from a plain text arguments file (in XML format) instead of reading them from the command line.

The exportdefcal utility only exports Control-M level Rule-Based Calendars. To export Folder level Rule-Based Calendars, use the exportdeffolder utility (as TAGs were previously exported).

Running the exportdeffolder utility

This procedure describes how to run the exportdeffolder utility, which enables you to export calendar definitions in the Control-M/EM database to an output file for use as input to other utilities.

➢ To run the utility:

1. Do one of the following:
   - **a.** Log on as a Control-M for Databases user.
   - **a.** Open a command prompt window (*Windows*). You do not need to be in the Control-M for Databases directory.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter one of the following commands:
   - `emdef exportdeffolder [-USERNAME <user> [-PASSWORD <password>] | -PASSWORD_FILE <passwordFile>] -HOST <GuiServerName> -ARG_FILE <argFileName> -OUT_FILE <outFileName>`
   - `emdef exportdeffolder [-u <user> [-p <password>] | -pf <passwordFile>] -s <GuiServerName> -arg <argFileName> -out <outFileName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the exportdeffolder parameters , see exportdefcal parameters (on page 281).

exportdefcal parameters

The following table lists the exportdefcal utility parameters:

Parameter	Description
<*userName*>	Control-M/EM user name.
<*password*>	The Control-M/EM user password.

<*passwordFile*>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName*>
	password=<*password*>

<*guiServerName*>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<*argFileName*>	Path and name of the arguments file containing exportdefcal specifications. Instructions for making this file are in XML file preparation (on page 616).
<*outFileName*>	Path and name of the file containing the specification of the exported job.
-ctm	Name of the Control-M installation that processes the job.
-folder	Name of the folder.
-app	Name of the application to which the job belongs.
-subapp	Name of the group to which the job belongs.

exportdefcal arguments file

Arguments are used as a selection criteria to determine which calendars to export. Arguments are written to the exportdefcal argument file. The arguments files that you create with the exportdefcal utility are written in XML format and saved in a text file. The format in which this file must be written is described on the following pages.

When this file is invoked, calendar definitions are exported from the Control-M/EM database. For instructions for creating arguments files, see XML file preparation (on page 616).

The following rules apply to the exportdefcal arguments file:

- More than one calendar can be specified in an exportdefcal file.
- The arguments file is case-sensitive.
- All parameter values (such as strings or digits) must be enclosed in quotation marks (for example, JOBNAME="Job1").
- More than one PARAM parameter can be used in a TERM statement.
- The relationship between PARAM parameters in a TERM statement is AND. The relationship between TERM statements is OR.

For more information on the input file parameters for the exportdefcal utility, see exportdefcal arguments file parameters (on page 282) and exportdefcal arguments file examples (on page 283).

exportdefcal arguments file parameters

The following table listst the exportdefcal arguments file parameters:

Parameter	Description	
The first two lines of the XML request file for this API request contain information that specifies the version of XML, the text encoding format being used, and the location of the .dtd file.		
TERMS	These tags indicate the start and end of the TERMS file. Only criteria that are located between the tags are considered to be part of the argument.	
TERM	The TERM tags indicate the start and the end of a group of selection criteria used to specify a calendar or calendars that are to be exported. Only PARAM tags that are located between the TERM tags are considered to be part of the TERM argument.	
	REL	Optional. Relationship between terms. Valid values: ▪ AND ▪ OR

Parameter	Description	
PARAM	The selection criteria parameter used to determine those calendars that are to be exported. More than one PARAM can be specified. Mandatory. `PARAM NAME="DATACENTER" OP="EQ" VALUE="Center1"`	
	NAME	String. Specify at least one of the following calendar parameter names. Valid values: ▪ **DATACENTER** ▪ **CALENDAR** ▪ **TYPE**
	OP	String. Mandatory. Describes the relationship between the NAME and the VALUE parameters of the TERM. Valid values: ▪ EQ ▪ NEQ ▪ NOTIN ▪ LIKE
	VALUE	String. Mandatory. Value of the parameter specified in the NAME field. ▪ If the value of NAME is DATACENTER, enter the name of the Control-M installation for VALUE. ▪ If the value of NAME is CALENDAR, enter a calendar name. ▪ If the value of NAME is TYPE, enter one of the following calendar types: Relative, Regular, Periodic, Rule_Based

exportdefcal arguments file examples

The following are examples of argument files used with the exportdefcal utility:

Exporting calendars

In this example, all calendars are exported in the data center named **Data1.** The output file contains all calendars in data center **Data1** that are named **Cal1.**

```
<TERMS>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>

<PARAM NAME="CALENDAR" OP="EQ" VALUE="CAL1"/>

</TERM>
```

```
</TERMS>
```

Exporting and importing a Rule-Based Calendar

In this example, an arguments file, **export_def_cal.arg** (shown in **export_def_cal.arg exportdefcal arguments file**), is used to export, all the Rule-Based Calendars that contain the "RuleBased" string in their names. The exportdefcal command (shown in **export command** is used to export the Rule-Based Calendar definition, RuleBasedCal, from the emdevA Control-M/EM GUI server to the **RuleBaseC.txt** export file. This export file is then used with the defcal command to import RuleBasedCal to the emdevB Control-M/EM GUI server.

export_def_cal.arg exportdefcal arguments file

```
<TERMS>

<TERM>

<PARAM NAME="CALENDAR" OP="LIKE" VALUE="RuleBased*"/>

<PARAM NAME="TYPE" OP="LIKE" VALUE="Rule_Based"/>

</TERM>

</TERMS>
```

export command

```
emdef exportdefcal -u userA -p passA -s emdevA -arg
\\netA\devpub\xargs\Arg\export_def_cal.arg -out
\\netA\devpub\xargs\Arg\RuleBaseC.txt
```

The RuleBaseC.txt XML output file

```
<?xml version='1.0' encoding='ISO-8859-1' ?>

<!DOCTYPE DEFCAL SYSTEM "defcal.dtd">

<DEFCAL >

 <RULE_BASED_CALENDAR

  ACTIVE_FROM="20101024"

  ACTIVE_TILL="20101021"

  APR="0"

  AUG="0"

  DATACENTER="dcDist_700"

  DAYS="1,2"

  DAYS_AND_OR="OR"

  DEC="0"

  FEB="0"

  JAN="0"

  JUL="0"

  JUN="0"

  MAR="0"
  MAXWAIT="00"
  MAY="0"
  NAME="RuleBasedCal"
  NOV="1"
  OCT="0"
  SEP="0"
  SHIFT="IGNOREJOB"
  SHIFTNUM="+00"
  WEEKDAYS="0"
/>
</DEFCAL>
```

import command

```
emdef defcal -u useB -p passB -s emdevB -src
\\netA\devpub\xargs\Arg\RuleBaseC.txt /o
```

updatedef

The updatedef utility updates (modifies) specified parameter values in the following definitions in the Control-M/EM database:

- Job processing definitions
- Folder definitions
- SMART Folder definitions
- Sub-folder definitions

- updatedef modifies the characteristics of existing job processing definitions.
- duplicatedefjob creates new job processing definitions based on existing job processing definitions in the "from" data center and folders.

The selected jobs, folders, SMART Folders and Sub-folders are modified according to specifications in the updatedef arguments file. The updatedef utility does not create new jobs or folders.

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Control-M/Server utilities

This table lists the Control-M/Server utilities for folders and calendars.

Utility Type	Description
ctmcalc_date (on page 286)	The ctmcalc_date utility calculates the date that a job can be ordered after adding or deducting a specified number of days.
ctmdeffolder (on page 289)	The ctmdeffolder utility defines a new SMART Folder. SMART Folders are used for jobs whose processing can be treated as a single unit.
ctmdefsubfolder (on page 292)	The ctmdefsubfolder utility defines a new Sub-folder.
ctmrpln (on page 295)	The ctmrpln utility creates a report that lists selected jobs in selected folders, and indicates when the jobs are scheduled to run.

ctmcalc_date

The ctmcalc_date utility calculates the date that a job can be ordered after adding or deducting a specified number of days. You can specify whether the calculated date can be any day of the week or must be a work day. To run the ctmcalc_date utility, see Running the ctmcalc_date utility (on page 286).

Running the ctmcalc_date utility

This procedure describes how to run the ctmcalc_date utility, which enables you to calculate the date that a job can be ordered after adding or deducting a specified number of days.

➢ To run the ctmcalc_date utility:

- Use the following command to invoke the ctmcalc_date utility:

```
ctmcalc_date -FOLDER <Folder Path name>
-NAME <job or sub folder name>
-DATE <scheduling date>
-SHIFT <+/-n>
-ONLY_WORKING_DAYS <Y/N>
-OUTPUT_FORMAT DATE | EXTENDED (Date and flag) | FLAG
[ -DEBUG    <debug level 0-5> ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmcalc_date parameters, see ctmcalc_date parameters (on page 287) and ctmcalc_date examples (on page 288).

ctmcalc_date parameters

The following table describes the parameters in the ctmcalc_date utility:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-FOLDER</td><td colspan="2">path of the SMART folder or sub-folder</td></tr>
<tr><td>-NAME</td><td colspan="2">name of the job or sub-folder</td></tr>
<tr><td>-DATE</td><td colspan="2">indicates the scheduling date (odate) to be associated with the job
Specify the date in <b>yyyymmdd</b> format.</td></tr>
<tr><td>-SHIFT</td><td colspan="2">indicates how many days to shift the scheduling criteria of the job
Valid values are <b>+n</b> or <b>-n</b>. <b>n</b> is the number of days to be shifted.
Specify <b>+n</b> to shift the job forward <b>n</b> number of days, or specify <b>-n</b> to shift the job backward <b>n</b> number of days. The scheduling criteria of the job are shifted either by work days or any day, according to the value specified for the -ONLY_WORKING_DAYS parameter.</td></tr>
<tr><td>-ONLY_WORKING_DAYS</td><td colspan="2">indicates whether the scheduling day can be any day of the week or must be a work day
Valid values are <b>Y</b> and <b>N</b>.
▪ <b>Y</b> – the job can be scheduled to run only on a work day
▪ <b>N</b> – the job can be scheduled to run on any day of the week</td></tr>
<tr><td rowspan="3">OUTPUT_FORMAT</td><td colspan="2">displays the following information:</td></tr>
<tr><td>DATE</td><td>when specified, displays the calculated date</td></tr>
<tr><td>EXTENDED</td><td>when specified, displays the calculated date and indicates if the job will be scheduled
Valid values are <b>Y</b> and <b>N</b>.
▪ <b>Y</b> – scheduled to run on the calculated date
▪ <b>N</b> – not scheduled to run on the calculated date</td></tr>
</table>

Parameter	Description	
	FLAG	when specified, indicates if the job will be scheduled Valid values are **Y** and **N.** ▪ **Y** – scheduled to run on the calculated date ▪ **N** – not scheduled to run on the calculated date
-DEBUG	activates a debug trace at the specified level Valid levels are 0–5. The default is **0** Performance is somewhat slower when operating in debug mode. BMC recommends that you activate debug mode only when requested by Customer Support.	

ctmcalc_date examples

The following are ctmcalc_date examples:

Issue the following command to display the calculated scheduling date that the prodyjob job in the Production folder will be ordered if its scheduling criteria are met, if two days are added to the original scheduling date of the job, August 02, 2008:

```
ctmcalc_date -FOLDER Production -NAME prodyjob -date 20080802 \
             -shift +2 -ONLY_WORKING_DAYS Y -OUTPUT_FORMAT DATE

20080804
```

As in the first example, displaying the calculated scheduling date and indicating if the job will be scheduled:

```
ctmcalc_date -FOLDER Production -NAME prodyjob -date 20080802 \
             -shift +2 -ONLY_WORKING_DAYS Y -OUTPUT_FORMAT EXTENDED

20080804 Y
```

As in the first example, indicating if the job will be scheduled without displaying the calculated scheduling date:

```
ctmcalc_date -FOLDER Production -MAME prodyjob -date 20080802 \
             -shift +2 -ONLY_WORKING_DAYS Y -OUTPUT_FORMAT FLAG

Y
```

ctmdeffolder

Use the ctmdeffolder utility to create a definition for a new SMART Folder. SMART Folders are used for jobs whose processing can be treated as a single unit. The definition created using this utility contains values for parameters that affect handling of the entire collection of jobs in the SMART Folder. A SMART Folder can be empty, or it can contain jobs and also Sub-folders, see ctmdefsubfolder (on page 292). To run the ctmdeffolder utility, see Running the ctmdeffolder utility (on page 289).

For more information about parameters of SMART Folders, see Job definition.

When a Sub-folder is defined using ctmdeftab without any Rule-Based Calendar being specified, the Sub-folder inherits all Rule-Based Calendars.

Running the ctmdeffolder utility

This procedure describes how to run the ctmdeffolder utility, which enables you to create a definition for a new SMART Folder.

➢ To run the ctmdeffolder utility:

- Do one of the following:
  - Use the following -input_file parameter to run the utility:

    ctmdeftab -input_file <*fullPathFileName*>
  - Use the following command to invoke the ctmdeffolder utility:

```
ctmdeftab
-FOLDER          <name>
 -SUB_APPLICATION          <sub_application name>
 -APPLICATION    <applic name>
 [ -ADJUST_COND  Y|N ]
 [ -RUN_AS        <username> ]
 [ -CREATED BY       <username> ]
 [ -DEBUG        <debug level 0-5> ]
 [ -QUIET  ]
 [ -TIMEZONE     <xxx> ]
 [ -TIMEFROM     <earliest submission time> ]
 [ -TIMEUNTIL    <latest  submission time> | '>' ]
 [ -PRIORITY     <job priority> ]
 [ -CONFIRM      Y|N ]
 [ -APPLTYPE     <agent_application> ]
 [ -APPLVER      <application version> ]
 [ -CMVER        <CM version> ]
 [ -APPLFORM     <application form> ]
 [ -DESCRIPTION  <string> ]
 [ -DOCMEM       <filename> ]
 [ -DOCLIB       <directory name> ]
 [ -INCOND       <condition> <dateref>|ODAT  AND|OR ]
 [ -OUTCOND      <condition> <dateref>|ODAT  ADD|DEL ]
 [ -VARIABLE     <varname> <expression> ]
 [ -SHOUT        OK|NOTOK|LATESUB|LATETIME|EXECTIME
                 <destination> <urgency R|U|V> <message> [<time>] ]
 [ -ORDER_ METHOD  <order method name>
 [ -ON         <OK|NOTOK>
     [ -DOOK ]
     [ -DONOTOK ]
     [ -DOSHOUT    <destination> <urgency R|U|V> <message> ]
     [ -DOCOND     <condname> <dateref>|ODAT  ADD|DEL ]
     [ -DOVARIABLE <varname> <expression> ]
     [ -DOFORCEJOB <foldername> <jobname> <odate>|ODAT ]
     [ -DOMAIL <destination> <cc> <urgency R|U|V> <subject> <message>
]
     [ -DOREMEDY <summary> <description> <urgency L|M|H|U|C> ]
-RBC <Rule_Based_Calendar_name>
     [ -MAXWAIT     <maxwait>        ]
     [ -DAYS        <daystr>         ]
     [ -WEEKDAYS    <weekdaystr>     ]
     [ -MONTH   ALL|JAN|FEB|MAR|APR|MAY|JUN|JUL|AUG|SEP|OCT|NOV|DEC
Y|N ]
     [ -DATE        <MMDD>           ]
     [ -DATEFROM    <YYYYMMDD>       ]
     [ -DATEUNTIL   <YYYYMMDD>       ]
     [ -DAYSCAL     <days calendar>  ]
     [ -WEEKCAL     <week calendar>  ]
     [ -CONFCAL     <conf calendar>  ]
     [ -CAL_ANDOR    AND|OR          ]
     [ -SHIFT        [</>/@][+/-]nn  ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmdeffolder parameters , see the following:

- ctmdeffolder parameters (on page 291)
- ctmdeffolder syntax rules (on page 292)
- ctmdeffolder example (on page 292)

ctmdeffolder parameters

The following table lists the ctmdeffolder utility parameters:

Parameter	Description
-DEBUG	Level of debug messages, 0 to 5. The default value is 0 (no debug messages.
-QUIET	If specified, no information messages are displayed during execution of the command.
-input_file	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: prepare and save files of utility parameters that can be reused. specify utility input longer than the number of characters allowed in the command line. `-input_file ~<controlmOwner>/ctm_server/data/ctmdeftab_parms.txt`

-ORDER_ METHOD	Name of a order method job that is associated with the created SMART Folder. This parameter is case-sensitive.
	The specified name must not be longer than 10 characters. If a longer name is specified, an error message is issued.
	`ctmdeftab -folder ss -application a -group g -rbc r -order_ method verylongnamespecified`
	The response is:
	`RULE-BASED CALENDAR 'r' added`
	The value length for -ORDER_ METHOD exceeds maximum allowed length 10

ctmdeffolder syntax rules

The following are the ctmdeffolder utility syntax rules:

- When using variables in cmtdeftab parameters, a variable that does not contain a $ sign can be enclosed in single (‘ ’) or double (" ") quotation marks.
- A variable that does contain a $ sign must be enclosed in single quotation marks.
- A variable containing a $ sign cannot be resolved if it is enclosed in double quotation marks.
- If you define a new Rule-based calendar with the ! character at the beginning of the Rule-based calendar name, the Rule-based calendar is excluded. If this feature is disabled, an error message is displayed that you cannot define a Rule-based calendar with the ! character. For more information, see **DefaultCTMExcludeRBC** in General parameters.

ctmdeffolder example

The following is an ctmdeffolder command example:

Use the following command to create a SMART Folder named job:

```
ctmdeftab  -FOLDER job -GROUP supply -APPLICATION supplies -RBC
jobRbc -DAYS ALL -MONTH ALL Y
```

Control-M/Server issues a message, similar to the following:

RULE-BASED CALENDAR 'jobRbc' added

new SMART Folder ENTITY defined, SMART Folder='job', ACTIVEFOLDERNO = 00000j(19)

ctmdefsubfolder

The ctmdefsubfolder utility creates a definition for a new Sub-folder. Sub-folders are used for jobs whose processing can be treated as a single unit. A Sub-folder can only be defined within a SMART folder, see ctmdeffolder (on page 289). A Sub-folder can be empty, or it can contain jobs and also other Sub-folders. To run the ctmdefsubfolder utility, see Running the ctmdefsubfolder utility (on page 292).

For more information about parameters of Sub-folders, see Sub Folder parameters.

Running the ctmdefsubfolder utility

This procedure describes how to run the ctmdefsubfolder utility, which enables you to create a definition for a new Sub-folder.

➢ To run the ctmdefsubfolder utility:

- Do one of the following:
  - Use the following -input_file parameter to run the utility:

    ctmdefsubtab -input_file <*fullPathFileName*>
  - Use the following command to invoke the ctmdefsubtab utility

```
ctmdefsubtab

-FOLDER            <name>
```

```
    -SUB_APPLICATION          <sub_application name>
    -APPLICATION    <applic name>
    [ -ADJUST_COND  Y|N ]
    [ -RUN_AS        <username> ]
    [ -CREATED BY       <username> ]
    [ -DEBUG        <debug level 0-5> ]
    [ -QUIET  ]
    [ -TIMEZONE     <xxx> ]
    [ -TIMEFROM     <earliest submission time> ]
    [ -TIMEUNTIL    <latest  submission time> | '>' ]
    [ -PRIORITY     <job priority> ]
    [ -CONFIRM      Y|N ]
    [ -APPLTYPE     <agent_application> ]
    [ -APPLVER      <application version> ]
    [ -CMVER        <CM version> ]
    [ -APPLFORM     <application form> ]
    [ -DESCRIPTION  <string> ]
    [ -DOCMEM       <filename> ]
    [ -DOCLIB       <directory name> ]
    [ -INCOND       <condition> <dateref>|ODAT  AND|OR ]
    [ -OUTCOND      <condition> <dateref>|ODAT  ADD|DEL ]
    [ -VARIABLE     <varname> <expression> ]
    [ -SHOUT        OK|NOTOK|LATESUB|LATETIME|EXECTIME
                    <destination> <urgency R|U|V> <message> [<time>] ]
    [ -ON         <OK|NOTOK>
        [ -DOOK ]
        [ -DONOTOK ]
        [ -DOSHOUT    <destination> <urgency R|U|V> <message> ]
        [ -DOCOND     <condname> <dateref>|ODAT  ADD|DEL ]
        [ -DOVARIABLE <varname> <expression> ]
        [ -DOFORCEJOB <foldername> <jobname> <odate>|ODAT ]
        [ -DOMAIL <destination> <cc> <urgency R|U|V> <subject> <message>
  ]
        [ -DOREMEDY <summary> <description> <urgency L|M|H|U|C> ]
  [ -RBC <rule_based_calendar_name|NONE|"*">    ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmdefsubfolder parameters, see ctmdefsubfolder parameters (on page 294) and ctmdefsubfolder syntax rules (on page 294).

ctmdefsubfolder parameters

The following table lists the ctmdefsubfolder utility parameters:

Parameter	Description
-DEBUG	Level of debug messages, 0 to 5. The default value is 0 (no debug messages).
-QUIET	If specified, no information messages are displayed during execution of the command.
-input_file	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: prepare and save files of utility parameters that can be reused. specify utility input longer than the number of characters allowed in the command line. `-input_file ~<controlmOwner>/ctm_server/data/ctmdefsubtab_parms.txt`

ctmdefsubfolder syntax rules

When using variables in ctmdefsubfolder parameters, a variable that does not contain a $ sign can be enclosed in single (‘ ’) or double (" ") quotation marks. A variable that does contain a $ sign must be enclosed in single quotation marks. A variable containing a $ sign cannot be resolved if it is enclosed in double quotation marks.

The RBC option associates Rule-Based Calendars to be used by the Sub-folder and it may appear more than once. Specified Rule-Based Calendars must be defined by the direct parent folder. The '*' means that the Sub-folder inherits all the Rule-Based Calendars from the direct parent folder. When a Sub-folder is defined without a Rule-based calendar, the Sub-folder inherits the Rule-based calendars of the direct parent. To define a Sub-folder without any Rule-based calendar associations, use the NONE option.If a Rule-based calendar is defined with the ! character at the beginning of the Rule-based calendar name, the Rule-based calendar is excluded. If this feature is disabled, an error message is displayed that you cannot define a Rule-based calendar with the ! character. For more information, see **DefaultCTMExcludeRBC** in General parameters.

The Created by field will be set automatically with the default logged in account name if it's not specified.

ctmrpln

The ctmrpln utility creates a report that lists selected jobs in selected folders, and indicates when the jobs are scheduled to run. It enables you to test the effect of different calendars on job scheduling. To run the ctmrpln utility, see Running the ctmrpln utility (on page 295).

Each report can be created in one of the formats described in ctmrpln report formats (on page 295).

The characters described in ctmrpln report characters (on page 296) can appear in this report. The characters indicate whether a job is scheduled to run (that is, whether the job is placed in the Active Jobs database.)

Running the ctmrpln utility

This procedure describes how to run the ctmrpln utility, which enables you to create a report that lists selected jobs in selected folders, and indicate when the jobs are scheduled to run.

➢ To run the ctmrpln utility:

- Type one of the following commands:
  - `ctmrpln` (and answer the prompts)
  - `ctmrpln <reportType> <calendar> <schedTab> <jobName> <date> [<output>]`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmrpln parameters, see ctmrpln parameters (on page 296) and ctmrpln examples (on page 297).

ctmrpln report formats

The following table lists the ctmrpln report formats:

Format	Description
Daily Report	Displays jobs in the specified folder that are scheduled to run on a specific day. Each job’s Mem Name (or Job Name), Group, and Description parameters are displayed.
Monthly Report	Displays a folder of all days in a specified month and marks (with an asterisk "*") the days of the month on which jobs in the specified folder are scheduled to run. Jobs can be identified either by their Mem Name or Job Name parameters.
Yearly Report	Displays the year, the two years before, and the two years after the year specified in the <Date> parameter. Marks each day with various characters (described in the next folder) that indicate if jobs in the specified folders are scheduled to run.

ctmrpln report characters

The following table lists the report characters that can be used with

Char	Description
*	The job is scheduled to run on this day.
.	The job is not scheduled to run on this day.
-	The job is scheduled to not run on this day. For example, if DAYS=-3, the job is scheduled to not run on the 3rd day of the month.

ctmrpln parameters

The following table lists the ctmrpln utility parameters:

Parameter	Description	
<ReportType>	Specify one of the following values:	
	DM (or D)	Creates a daily report, identifying each job according to its Mem Name parameter.
	DJ	Creates a daily report, identifying each job according to its Job Name parameter.
	MM (or M)	Creates a monthly report, identifying each job according to its Mem Name parameter.
	MJ	Creates a monthly report, identifying each job by its Job Name parameter.
	Y	Creates a yearly report.
<calendar>	Specify one of the following values:	
	Y	Creates the report using the calendar specified in the job's scheduling parameters.
	N	Creates the report ignoring the calendar specified in the job's scheduling parameters.
	<Name>	Creates the report using a calendar you specify (ignoring calendars specified in the job scheduling parameters). Use this option to see how another calendar affects scheduling.

Parameter	Description
<schedTab>	Name of the folder on which to base the report. The folder name can include the following wildcard characters:
	* represents any number of characters (including no characters). Specify * by itself to include all folders. Any parameter including * should be enclosed in quotation marks.
	? represents any single character.
<jobName>	Job Name of the jobs to include in the report. The Job Name can include wildcard characters (see <Schedtab**>** above). Specify * by itself to include all jobs in the folder.
<date>	Date for the report: For daily reports: A date in yyyymmdd format. For monthly reports: A month in yyyymm format. For yearly reports: A year in yyyy or yy format. The ctmrpln utility supports years from 1972 to 2035, inclusive.
<output>	Full path name to which the report should be sent (optional). If this parameter is not specified, the output is routed to the default output device. To print the Monthly Report, specify a device that can print 132-column reports.

ctmrpln examples

The following are examples of the ctmrpln utility:

The following command causes the utility to generate a report for folder PROD1. The report will include all jobs whose Job Name parameter begins with "jn" and that will run on January 1, 2008 based on the calendar work_days. The job is identified by its Mem Name. (To identify jobs by Job Name, specify Report_type DJ.) The output is directed to the user's display.

```
ctmrpln D work_days PROD1 "jn*" 20080101
```

The following command causes the utility to generate a folder of days on which job PRDJ02 in folder PROD1 will run during the month of April, 2008, based on the calendar work_days. The job is identified by Mem Name. (To identify jobs by Job Name, specify Report_type MJ.) The output is directed to printer lp1.

```
ctmrpln M work_days PROD1 PRDJ02 200804 lp1
```

The following command causes the utility to generate a five-year report encompassing the period January 2003 through December 2008. This indicates on which days each job in folder PROD1 runs, based on the calendar work_days. The output is directed to printer lp1.

```
ctmrpln Y work_days PROD1 "*" 2005 lp1
```

exportdeffolder

The exportdeffolder utility exports folders from the Control-M/EM database to a file.

When the exportdeffolder utility is invoked, a file of arguments that you have created is processed. This arguments file contains statements that specify an existing folder, SMART Folders and Sub-folders. The specified folders are exported to an output file. For more information, see The exportdeffolder arguments file (on page 300).

Output files created with the exportdeffolder utility can be used as import files with the deffolder utility.

For example, you can export job processing definitions to an output file using exportdeffolder, make modifications to the definitions and save the file, and use the same file as the input file when running deffolder to import the modified folder definitions into Control-M/EM database.

exportdeffolder parameters

The following table describes the exportdeffolder utility parameters:

Parameter	Description
<*userName*>	Control-M/EM user name.
<*password*>	The Control-M/EM user password.

<*passwordFile*>	Flat file containing an unencrypted user name and password on separate lines in the format:
	user=<*userName*>
	password=<*password*>

<*guiServerName*>	Control-M/EM GUI server logical name, host name, or IP address. If multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.
<*argFileName*>	Path and name of the arguments file containing the exportdeffolder specifications. Instructions for preparing this file are in XML file preparation (on page 616)
<*outFileFme*>	Path and name of the file containing the specification of the exported job.
-ctm	Name of the Control-M installation that processes the job
-folder	Name of the folder.
-app	Name of the application to which the job belongs.
-subapp	Name of the group to which the job belongs.

The exportdeffolder arguments file

Arguments are used as a s election criteria to determine which folders to export. Arguments are written to the exportdeffolder argument file.

The arguments files that you create with the exportdeffolder utility are written in XML format and saved in a text file. The format in which this file must be written is described on the following pages.

When this file is invoked, folder definitions are exported from the Control-M/EM database. For instructions for creating arguments files, see XML file preparation (on page 616).

The following rules apply to the exportdeffolder arguments file:

- More than one job can be specified in an **exportdeffolder** file.
- The arguments file is case-sensitive.
- All parameter values (such as strings or digits) must be enclosed in quotation marks (for example, JOBNAME="Job1").
- More than one PARAM parameter can be used in a TERM statement.
- The relationship between PARAM parameters in a TERM statement is AND. The relationship between TERM statements is OR.

The exportdeffolder arguments file is checked and processed. If there are any errors in the file, a message is displayed specifying the lines with errors.

The exported folder definitions are saved to an output file, the name and location of which is specified in the *outFileName* parameter.

For more information on the arguments file parameters for the exportdeffolder utility, see exportdeffolder arguments file parameters (on page 300) and exportdeffolder examples (on page 302).

exportdeffolder arguments file parameters

The following table lists the exportdeffolder arguments file parameters:

Parameter	**Description**	
The first two lines of the XML request file for this API request contain information that specifies the version of XML, the text encoding format being used, and the location of the **.dtd** file.		
TERMS	These tags indicate the start and end of the TERMS file. Only criteria that are located between the tags are considered to be part of the argument.	
TERM	The TERM tags indicate the start and the end of a group of selection criteria used to specify a folder or folders that are to be exported. Only PARAM tags that are located between the TERM tags are considered to be part of the TERM argument.	
	REL	Optional. Relationship between terms. Valid values: ▪ AND ▪ OR

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="4"><b>PARAM</b></td><td colspan="2">The selection criteria parameter used to determine those folders that are to be exported. More than one PARAM can be specified. Mandatory.
`PARAM NAME="DATACENTER" OP="EQ" VALUE="Center1"`</td></tr>
<tr><td><i>NAME</i></td><td>String. Mandatory.
The parameter name of any folder or SMART Folder parameter. These parameters are described in deffolder (on page 257).
At least one of the following folder parameters must be included in the arguments file: DATACENTER, FOLDER_NAME, FOLDER_DSN</td></tr>
<tr><td><i>OP</i></td><td>Mandatory.
Describes the relationship between the NAME and the VALUE parameters of the TERM. Valid values:
- EQ
- NEQ
- NOTIN
- LIKE</td></tr>
<tr><td><i>VALUE</i></td><td>String. Mandatory.
The value of any folder, SMART Folder or Sub-folder parameter. These parameters are described in deffolder (on page 257).
Multiple values can be specified for VALUE by using the * wildcard character in place of characters at the end of an expression.</td></tr>
<tr><td><i>CYCLIC_TYPE</i></td><td colspan="2">Determines the type of cyclic job:
- Interval
- IntervalSequence
- SpecificTimes
Relevant for Control-M version 6.4.01 or later.</td></tr>
<tr><td><i>CYCLIC_TOLERANCE</i></td><td colspan="2">Maximum delay in minutes permitted for a late submission when selecting a specific time (for example 5 minutes).
Relevant for Control-M version 6.4.01 or later.</td></tr>
<tr><td><i>CYCLIC_INTERVAL_SEQUENCE</i></td><td colspan="2">A list of time intervals, separated by commas, (for example +30M,+2H,+1D) up to 4000 characters including commas.
Relevant for Control-M version 6.4.01 or later.</td></tr>
</table>

Parameter	Description
CYCLIC_TIMES_ SEQUENCE	A list of times, separated by commas (for example 0800,1330,2300), which supports time synonym (for example 2730). Relevant for Control-M version 6.4.01 or later.

exportdeffolder examples

Following are examples of arguments files used with the exportdeffolder utility:

Export all folders in the Data1 data center

```
<TERMS>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>

</TERM>

</TERMS>
```

Export with multiple selection criteria

Exported SMART Folders that:

are located in data center **Data1** and belong to the GRP_**03** SMART Folder

- or -

are located in data center **Data1** and belong to the GRP_**04** SMART Folder

```
<TERMS>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>

<PARAM NAME="GROUP" OP="EQ" VALUE="GRP_03"/>

</TERM>

<TERM>

<PARAM NAME="DATACENTER" OP="EQ"
VALUE="Data1"/>

<PARAM NAME="GROUP" OP="EQ" VALUE="GRP_04"/>

</TERM>

</TERMS>
```

Uploading, ordering and scheduling jobs and folders using the cli utility

The Command Line Interface (cli) utility is a batch utility that enables you to perform the following operations (services) from the command line:

- Upload or download folders
- Order or force folders
- Order or force jobs
- Force jobs in a folder
- Upload or download calendars
- Delete job processing definitions from folders

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

cli

The cli utility can be used on UNIX and Windows computers. The cli utility is installed automatically on Windows computers during installation of the Control-M/EM Gateway, GUI server, and Control-M Configuration Manager components. To run the cli utility, see Running the cli utility (on page 303).

Many of the tasks performed by the cli utility can also be performed using Control-M for Databases and Control-M Workload Automation. However, by including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

You can make multiple requests in a single operation. Each service requires its own service name and includes all the relevant service parameters that follow it.

Running the cli utility

This procedure describes how to run the cli utility, which enables you to run Control-M/EM commands.

Before you begin

- You must have proper security authorization for any of the actions you perform using cli.

➢ To run the cli utility:

1. Change the working directory to the Control-M for Databases home directory.
2. Enter the following command, depending on your operating system:

- **On Windows**

  ```
  cli [{(-U emUser -P emPass) | -pf passwordFile}] -h hostName
  [-t timeout] [-DDMM] [-BY_FORCE] <cmd> <cmd> [ACTION_REASON <reason for
  taking an audit action> [ACTION_NOTE <descriptive reason for audit
  action>]...
  ```

- **On UNIX**

  ```
  em cli [{(-U <emUser> -P <emPass>) | -pf <passwordFile>}] -h <hostName>
  [-t <timeout>] [-DDMM] [-BY_FORCE]
  <cmd> <cmd> [ACTION_REASON <reason for taking an audit action>
  [ACTION_NOTE <descriptive reason for audit action>]...
  ```

The prefix **em** must be replaced with the prefix **ecs** when using the cli utility on Control-M for Databases versions earlier than 7.0.00.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see cli parameters (on page 305) and Notes applicable to cli commands (on page 307).

cli parameters

The following table lists the cli utility parameters:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>*<emUser>*</td><td colspan="2">Control-M for Databases user name.</td></tr>
<tr><td>*<emPass>*</td><td colspan="2">Control-M for Databases user password.</td></tr>
<tr><td>*<passwordFile>*</td><td colspan="2">Flat file containing an unencrypted user name and password on separate lines in the format:
user=*<emUser>*
password=*<emPass>*</td></tr>
<tr><td>*<hostName>*</td><td colspan="2">Host name of the workstation running Control-M for Databases Server.
If you need to address a GUI server and multiple GUI servers exist, set this parameter to the logical name of the relevant GUI server.</td></tr>
<tr><td>*<timeout>*</td><td colspan="2">Time, in seconds, that the utility waits for a response. Timeout is used to override the default waiting period (120 seconds).
Do not use timeout with commands that do not return a response (-JOB_DELETE and -MEM_DELETE).</td></tr>
<tr><td>-DDMM</td><td colspan="2">If specified, reverses the Odate format, as described below.</td></tr>
<tr><td>-BY_FORCE</td><td colspan="2">Forces the specified folder or calendar. Use this option during upload only.</td></tr>
<tr><td>-ACTION_REASON</td><td colspan="2">A note saved in the audit report explaining the purpose for performing the action.</td></tr>
<tr><td>-ACTION_NOTE</td><td colspan="2">A note saved in the audit report justifying the performance of the action.</td></tr>
<tr><td>*<cmd>*</td><td colspan="2">The syntax for additional commands that are available for specifying with the cli utility are shown below.</td></tr>
<tr><td></td><td>-JOB_ORDER</td><td><CTM_name> <folderName> <job_name> <Odate> <Wait_Odate>[<With_Hold>] [<library>]</td></tr>
<tr><td></td><td>-JOB_FORCE</td><td><CTM_name> <folderName> <job_name> <Odate><Wait_Odate> [<With_Hold>] [<library>]</td></tr>
</table>

Parameter	Description	
	-JOB_ORDER_INTO	<CTM_name> <folderName> <job_name> <folder> [<Duplication>] <Odate><Wait_Odate> [<With_Hold>] [<library>]
	-JOB_FORCE_INTO	<CTM_name> <folderName> <job_name> <folder> [<Duplication>] <Odate> <Wait_Odate> [<With_Hold>] [<library>]
	-SUB_FOLDER_FORCE_INTO	<CTM_name> <sub-folderName> <folder> [<Duplication>] <Odate> <Wait_Odate> [<With_Hold>] [<library>]
	-FOLDER_ORDER	<CTM_name> <folderName> <Odate> <Wait_Odate> [<With_Hold>][UNIQUE_FLOW] [<library>]
	-FOLDER_FORCE	<CTM_name> <folderName> <Odate> <Wait_Odate>[<With_Hold>][UNIQUE_FLOW] [<library>]
	-FOLDER_UPLOAD	<CTM_name> <folderName> [<library>]
	-FOLDER_DOWNLOAD	<CTM_name> <folderName> [<library>]
	-CAL_UPLOAD	<CTM_name> <calendar_name>
	-CAL_DOWNLOAD	<CTM_name> <calendar_name>
	-SUB_FOLDER_DELETE	<CTM_name> <sub_folderName>
	-JOB_DELETE	<CTM_name> <folderName> <job_name> ALL|NONE|NUMBER
	-MEM_DELETE	<CTM_name> <folderName> <mem_name> ALL|NONE|NUMBER [<library>]

For more information on the sub elements, see Notes applicable to cli commands (on page 307).

Notes applicable to cli commands

The following notes are applicable to the cli parameters (on page 305):

Command	Description
<Odate>	Specify either as: MMDD or YYYYMMDD. <Odate> may also be ODAT for Control-M version 6.0.00 or later (Order or Force Folder in the Original Scheduling Date) If the -DDMM is specified, <Odate> can be specified as DDMM or DDMMYYYY
<folder>	Valid values are: ▪ RECENT - Force Job into the recent SMART Folder that was previously ordered ▪ NEW– Force Job into a new folder ▪ STANDALONE – Force Job as a standalone job
<Sub_application OrderID>	Force the specified job into the specified folder.
<Duplication>	Specify if <folder> is RECENT or <GroupOrderID>, otherwise do not specify. Specify one of the following values: ▪ N - Don't allow duplication of the job. ▪ Y - Allow duplication of the job.
<Wait_Odate>	Determines whether you should wait for the Odate to run the job. ▪ For Control-M for Z/OS this field is Mandatory. Valid values are: Y - Wait for Odate to run job. N - Run the job immediately. For Control-M for Distributed Systems this field is optional. If omitted,the job runs immediately. If not omitted, the only valid value is Wait_Odate, which means wait for Odate to run job.
<With_Hold>	Relevant only for Control-M for z/OS version 6.2.00 or later and for Control-M for Distributed Systems version 6.3.00 or later. Specify one of the following values: ▪ N - Order/Force the job in a free state. ▪ Y or With_Hold - Order/Force the job in a Hold state.

Command	Description
<library>	Mandatory when ordering MVS Jobs.

Uploading and downloading folders

This procedure describes a series of commands that enable you to perform various tasks. Sub-folders cannot be uploaded or downloaded independently.

➢ To upload/download a folder:

1. To upload a folder, type one of the following commands depending on your operating system:
   - **On Windows**

     ```
     cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName
     [-t timeout] -FOLDER_UPLOAD control-mFolder [Odat] [library]
     ```

   - **On UNIX**

     ```
     em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
     [-t <timeout>] -FOLDER_UPLOAD <control-mFolder> [Odat] [library]
     ```

2. To download a folder, type one of the following commands depending on your operating system:
   - **On Windows**

     ```
     cli [{(-U dbUser -P dbPass) | -pf passwordFile}]
     -h hostName [-t timeout] -FOLDER_DOWNLOAD control-mFolder [library]
     ```

   - **On UNIX**

     ```
     em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}]
     -h <hostName> [-t <timeout>] -FOLDER_DOWNLOAD <control-mFolder>
     [library]
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see cli parameters (on page 305) and Notes applicable to cli commands (on page 307).

Forcing folders

This procedure describes how to force a folder. Sub-folders cannot be forced, except into a SMART Folder.

➢ To force a folder:

- Type one of the following commands, depending on your opertaing system:
  - **On Windows**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName [-t timeout]
    [-DDMM] -FOLDER_FORCE control-mFolder [Odate] [library]
    ```

  - **On UNIX**

```
em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
[-t timeout] [- DDMM] -FOLDER_FORCE <control-mFolder> [Odate] [library]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see cli parameters (on page 305) and Notes applicable to cli commands (on page 307).

Forcing Sub-folders into a SMART folder

The following procedure describes how to force a Sub-folder into a SMART Folder.

➢ To force a Sub-folder into a SMART folder:

- Type one of the following commands, depending on your operating system:
  - **On Windows**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName [-t timeout]
    [-DDMM] -SUB-FOLDER_FORCE_INTO <CTM_name> sub_folder_name> <folder>
    [<Duplication>]<Odate> <Wait_Odate> [<With_Hold>]
    ```

  - **On UNIX**

    ```
    em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
    [-t <timeout>] [-DDMM] -SUB-FOLDER_FORCE_INTO <CTM_name>
    sub_folder_name> <folder> [<Duplication>]<Odate> <Wait_Odate>
    [<With_Hold>]
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see the following:

- cli parameters (on page 305)
- Notes applicable to cli commands (on page 307)
- Force Sub-folder into SMART Folder parameters (on page 309)

Force Sub-folder into SMART Folder parameters

The following table lists the parameters for the force Sub-folder into SMART folder commands:

Parameters	Description	
Folder	SMART Folder into which the Sub-folder is forced. Valid values:	
	RECENT	Forces the Sub-folder into the SMART Folder that was run most recently.
	NEW	Creates a new folder.

<table>
<tr><th>Parameters</th><th colspan="2">Description</th></tr>
<tr><td></td><td>smart_folderID</td><td>SMART Folder into which the Sub-folder is forced.</td></tr>
<tr><td>[duplication]</td><td colspan="2">Adds a Sub-folder to a SMART Folder, even if there is a Sub-folder with that name in the SMART Folder. Valid values:
▪ Y – Adds the Sub-folder, if required.
▪ N – Does not create a duplicate Sub-folder if a Sub-folder of the same name already exists
This setting can be used only when RECENT or smart_folderID are selected for folder.</td></tr>
</table>

Ordering folders

This procedure describes how to order folders.

➢ To order a folder:

- **Type one of the following commands, depending on your operating system:**
  - **On Windows:**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName [-t timeout]
    [-DDMM] -FOLDER_ORDER control-mFolder [Odate] [library]
    ```

  - **On UNIX**

    ```
    em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
    [-t <timeout>] [-DDMM] -FOLDER_ORDER <control-mFolder> [Odate]
    [library]
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see cli parameters (on page 305) and Notes applicable to cli commands (on page 307).

Deleting Sub-folders

This procedure describes how to delete Sub-folders using the cli utility which enables you to delete the Sub-folder from the Control-M/EM database only. Sub-folders in the Control-M/Server database are not affected.

➢ To delete Sub-folders:

- Type oe of the following commands depending on your platform:
  - **On Windows**:

```
cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName
-SUB-FOLDER_DELETE <control-m Name> <sub-folderName>
```

- **On UNIX**:

  ```
  em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
  -SUB-FOLDER_DELETE <control-m Name> <sub-folderName>
  ```

Deleting Sub-folders using the cli utility will delete all of the Sub-folders contents. If you only want to delete a job use the -JOB_DELETE command.

Forcing a job

The following procedure describe how to force jobs.

➢ To force a job:

- Type one of the following commands, depending on your operating system:
  - **On Windows**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName [-t timeout]
    [-DDMM] -JOB_FORCE control-mFolder jobName [Odate] [library]
    ```

  - **On UNIX**

    ```
    em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}]  -h <hostName>
    [-t <timeout>] [-DDMM] -JOB_FORCE <control-mFolder> <jobName> [Odate]
    [library]
    ```

For more information on the forcing a job parameters, see Force a job parameters (on page 312).

Forcing a job into a folder

The following procedure describes how to force a job into a folder.

➢ To force a job into a folder:

- Type one of the following commands, depending on your operating system:
  - **On Windows**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName [-t timeout]
    [-DDMM] -JOB_FORCE_INTO control-mFolder jobName folder [duplication]
    [Odate] [library]
    ```

  - **On UNIX**

    ```
    em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
    [-t <timeout>] [-DDMM] -JOB_FORCE_INTO <control-mFolder> <jobName>
    folder [duplication] [Odate] [library]
    ```

For more information on the forcing a job parameters, see Force a job parameters (on page 312).

Force a job parameters

The following table lists the parameters for forcing a job commands:

Parameters	Description	
Folder	Folder into which the job is forced. Valid values:	
	RECENT	Forces the job into the folder that was run most recently.
	NEW	Creates a new folder.
	STAND ALONE	Forces the job without adding it to a folder.
	SUB-APPLICATION	Folder into which the job is forced.
[duplication]	Adds a job to a folder, even if there is a job with that name in the folder. Valid values: ▪ Y – Adds the job, if required. ▪ N – Does not create a duplicate job if a job of the same name already exists This setting can be used only when RECENT or GROUPID are selected for folder.	

Ordering jobs

The following procedure describes how to order jobs

➢ To order a job:

- Type one of the following commands depending on your operating system:
  - **On Windows**

```
cli [-U <userName> [-P <password>] | -PF <passwordFile>] -H <serverName>
[-T <timeoutInSeconds>] [-DDMM] [-BY_FORCE] -JOB_ORDER <ctmName>
<folderName> <jobName> <Odate> <waitOdate> [<withHold>] [<library>]
```

  - **On UNIX**

```
em cli [-U <userName> [-P <password>] | -PF <passwordFile>] -H
<serverName>
[-T <timeoutInSeconds>] [-DDMM] [-BY_FORCE] -JOB_ORDER <ctmName>
<folderName> <jobName> <Odate> <waitOdate> [<withHold>] [<library>]
```

When forcing or ordering a job, Control-M/EM does not check if multiple jobs with the same name exist in the folder.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see cli parameters (on page 305) and Notes applicable to cli commands (on page 307).

Deleting job definitions by Job Name

The following procedure describes how to delete a job definitions using the cli utility. This deletes the job definitions from the Control-M/EM database only. Job definitions in the Control-M/Server database are not affected.

➢ To delete a job definition by Job Name:

- Type one of the following commands depending on your operating system:
  - **On Windows**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName -JOB_DELETE
    control-mFolder jobName delete_type [library]
    ```

  - **On UNIX**

    ```
    em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
    -JOB_DELETE <control-mFolder> <jobName> delete_type [library]
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see the following:

- cli parameters (on page 305)
- Notes applicable to cli commands (on page 307)
- Delete a job definition parameters (on page 314)

Deleting job definition by File Name

The following procedure describes how to delete a job definitions using the cli utility. This deletes the job definitions from the Control-M/EM database only. Job definitions in the Control-M/Server database are not affected.

➢ To delete a job definition by File Name:

- Type one of the following commands depending on your operating system:
  - **On Windows**

    ```
    cli [{(-U dbUser -P dbPass) | -pf passwordFile}] -h hostName -MEM_DELETE
    control-mFolder memName delete_type [library]
    ```

  - **On UNIX**

    ```
    em cli [{(-U <dbUser> -P <dbPass>) | -pf <passwordFile>}] -h <hostName>
    -MEM_DELETE <control-mFolder> <memName> delete_type [library]
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the cli parameters, see the following:

- cli parameters (on page 305)
- Notes applicable to cli commands (on page 307)
- Delete a job definition parameters (on page 314)

Delete a job definition parameters

The following table lists the parameters for deleting a job definition commnads:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="3">delete_type</td><td colspan="2">Indicates the type of operation to be performed. Valid values:</td></tr>
<tr><td>ALL</td><td>All occurrences of the job are deleted, if there is more than one job with the same name.</td></tr>
<tr><td>NONE</td><td>No jobs are deleted if there is more than one job with the same name.</td></tr>
<tr><td>sequence_number</td><td colspan="2">Deletes the job with the specified sequence number of the duplicate job (for example, if 5 is entered, the fifth occurrence of the Job Name is deleted).</td></tr>
<tr><td>Library</td><td colspan="2">Required for z/OS job definitions.</td></tr>
</table>

New day procedure and order methods Control-M/Server utilities

The new day procedure and order methods utilities:

- Obtain information regarding the user daily procedure, such as the last time it ran or the jobs that it ordered and their security authorizations
- Actually run a specific order method to order jobs whose folders are associated with it (ctmudly)

By including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utility	Description
ctmordck (on page 316)	Lists job processing definitions that can be ordered by a specific Order method job.
ctmudchk (on page 318)	Allows recovery from interruption of a Order method.
ctmudlst (on page 320)	Enables you to manually set the Order method last run date.
ctmudly (on page 321)	Orders jobs for a specific Order method name.

ctmordck

The ctmordck utility lists job processing definitions associated with a specific Order method name and indicates the security status of each job with regard to the owner of the Order method job (that is, whether or not the Control-M security mechanism will allow jobs associated with a Order method name to run with the authorizations currently assigned to the owner of the Order method job). To run the ctmordck utility, see Running the ctmordck utility (on page 316).

This utility displays the following fields:

- Name of the job.
- Number of the job in the Control-M/Server database.
- Author of the job processing definition.
- Owner of the job processing definition.

The following information is displayed for each job processing definition:

- **TB**: Whether or not the owner of the Order method job is authorized to order the folder of the job listed.
- **FL**: Whether or not the owner of the Order method job is authorized to execute the script file of the job listed.
- **UA**: Whether or not the owner of the Order method job is authorized to order jobs for the owner of the job listed.

This utility can be used non-interactively for non-terminal destinations, see the description of the <Output> parameter in ctmordck parameters (on page 317).

When the command **ctmordck** is invoked, prompts are displayed on the screen to assist you to specify the parameters of this utility.

Running the ctmordck utility

This procedure describes how to run the ctmordck utility, which enables you to list job processing definitions associated with a specific Order Method name and indicates the security status of each job with regards to the owner of the Order Method job.

➢ To run the ctmordck utility:

- Type either of the following commands:
  - `ctmordck <User Name> <Order method> [<Output>]`
  - `ctmordck`

For more information on the ctmordck parameters, see ctmordck parameters (on page 317).

ctmordck parameters

The following table lists the ctmordck utility parameters:

Parameter	Description
<User Name>	Owner of the Order method job.
<Order method>	Name of the Order method procedure.
<Output>	Full path name to which the report should be sent. If this parameter is not specified, the output is routed to the default output device.

ctmordck examples

The following are examples of ctmordck utility command and output:

The following command generates a list for user SYSTEM and the Order method SYSTEM. The list is directed to the specified file called udlist.

```
ctmordck SYSTEM SYSTEM ~<controlm_owner>/ctm_server/user1/udlist
```

The following is sample output from the above command:

```
Date: 10-NOV-2007.   Page: 1
   User SYSTEM , Daily SYSTEM Ordering list

JOBNAME     No.  AUTHOR  OWNER  TB  FL  UA

----------  ---  ------  -----  --  --  --

CTMLOG HAN  2066  root    root   Y   N   Y

PURGE JOB   2067  root    root   Y   N   Y

user3-DAIL  2033  BARRY   user3  Y   N   Y

user2-DAIL  2032  STEVE   user2  Y   N   Y

user1-DAIL  2031  STEVE   user1  Y   N   Y

JEAN-UD     2000  jean    jean   Y   N   Y

JOB-STATUS  2068  root    root   Y   N   Y

GD-TEST1      20  jean    user1  Y   N   Y

GD-TEST2      21  jean    user2  Y   N   Y

GD-TEST3      22  jean    user3  Y   N   Y

GD-user4    2008  jean    user4  Y   N   Y

GD-user5    2009  jean    user5  Y   N   Y
```

ctmudchk

The ctmudchk utility checks if all jobs that should have been ordered by a Order Method job are in the Active Jobs database. This utility facilitates recovery from the interruption of a Order Method job. To run the ctmudchk utility, see Running the ctmudchk utility (on page 318).

When using the ctmudchk utility, the New Day procedure must not be running (the status of the data center in the Communication Status window of Control-M/EM must not be "Formatting AJF").

Running the ctmudchk utility

This procedure describes how to run the ctmudchk utility, which enables you to check if all jobs that should have been ordered by a Order Method job are in the Active Jobs database.

➢ To run the ctmudchk:

1. Use the following command to run the ctmudchk utility:

```
ctmudchk
  -DAILY         <order method name>
  -ACTION        <LIST|ORDER>
  [ -ODATE         <scheduling date>     ]
  [ -ODATE_OPTION <VALUE_DATE|RUN_DATE> ]
  [ -FILE          <file name>           ]
```

2. Check return codes.

The utility returns code 1 status (NOTOK) if it attempts to order a job, but fails to do so. Otherwise, the utility returns code 0 status (OK).

For more information on ctmudchk parameters, see ctmudchk parameters (on page 318).

ctmudchk parameters

The following table lists the ctmudchk utility parameters:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-DAILY</td><td colspan="2">Name of the Order method to be checked.</td></tr>
<tr><td rowspan="3">-ACTION</td><td colspan="2">Indicates whether jobs that are missing from the Active Jobs database should be listed or ordered.
The following values can be specified for this parameter:</td></tr>
<tr><td>LIST</td><td>Lists the job name and the name of the folder for each missing job.</td></tr>
<tr><td>ORDER</td><td>Orders the missing jobs.</td></tr>
</table>

Parameter	Description	
-odate	Indicates the scheduling date (odate) to be associated with jobs that are ordered by this order method job. Valid values are:	
	ODAT	The current working date of the computer on which Control-M/Server is running. This is the default value.
	yyyymmdd	A specific working day in yyyymmdd format.
	The interpretation of this parameter value is dependent on the value specified for the -odate_option parameter (described below).	
-odate_option	Indicates how the specified -odate value should be used. Valid values are:	
	value_date	The specified odate is the odate value for the jobs. However, the jobs ordered by the order method job should be run during the current working day. This is the default value for the -odate_option parameter. If time zones specified in specific job processing definitions in the folders, then the jobs are run according to those time zones.
	run_date	The jobs ordered by the order method job should be run only when the specified odate begins. If the specified odate is the current working day, then this will work in the same way as value_date (described above). If the specified odate has not begun (for example, due to time zone specifications), then the job will wait in the Active Jobs database (with WAIT_ODAT status) until the start of the specified working day. If the specified odate has already passed, the ctmudchk utility will not run, and an error message will be displayed.
-FILE	Indicates the path name for the output of the ctmudchk utility. This parameter is required only if LIST is specified for the ACTION parameter.	

ctmudchk examples

The following are exampled of ctmudchk utility commands:

Use the following command to check the Active Jobs database for jobs that are ordered by the Order method whose name is payroll. The Job Name and Folder are listed for each job that is not in the Active Jobs database.

```
ctmudchk -DAILY payroll -ACTION
```

Use the following command to check the Active Jobs database for jobs that are ordered by the Order method whose name is admin1. The utility orders each job that is not in the Active Jobs database.

```
ctmudchk -DAILY admin1 -ACTION ORDER
```

ctmudlst

The ctmudlst utility is used to display or modify UDLAST (the Order method last run date). To run the ctmudlst utility, see Running the ctmudlst utility (on page 320).

When using the ctmudlst utility, the New Day procedure must not be running (that is, the status of the data center in the Communication Status window of Control-M/EM must not be "Formatting AJF").

Running the ctmudlst utility

This procedure describes how to run the ctmudlst utility, which enables you to display or modify UDLAST (the Order Method last run date).

➢ To run the ctmudlst utility:

- Specify one of the following commands:
  - `ctmudlst LIST  <order_ method>`
  - `ctmudlst LIST  "*"`
  - `ctmudlst UPDATE  <order_ method> <date>`

For more information on the ctmudlst utility parameters, see ctmudlst parameter (on page 321).

ctmudlst parameter

The following table lists the ctmudlst utility parameters:

Parameter	Description
LIST	Lists the Order method last run date.
UPDATE	Updates the Order method last run date.
<Order method>	Order method name.
"*"	Asterisk enclosed in quotation marks. Displays a list of all Order method names and corresponding last run dates.
<Date>	Requested value for the last running date in yyyymmdd format.

ctmudlst examples

The following are examples of the ctmudlst utility commands:

The following command lists the last run date for Order method payroll:

```
ctmudlst LIST payroll
```

The following command changes the last run date for Order method inventory to August 10, 2008:

```
ctmudlst UPDATE inventory 20080810
```

ctmudly

The ctmudly utility orders jobs whose folders are associated with a specific Order Method name. To run the ctmudly utility, see Running the ctmudly utility (on page 321).

Each job in the ordered Folders whose Scheduling criteria are satisfied is placed in the Active Jobs database.

The exit code of the ctmudly utility is determined by Control-M configuration parameter UDLY_PARTCOPY_ERR. For more information about this parameter, see Configuration parameters.

Running the ctmudly utility

This procedure describes how to run the ctmudly utility, which enables you to order jobs whose folders are associated with a specific Order Method name.

➢ To run the ctmudly utility:

1. Type the following command:

```
ctmudly -DAILY_NAME <order_ method>
```

```
        [-odate {ODAT|<yyyymmdd>}]
        [-odate_option {value_date|run_date}]
```

2. Check for messages issued when a job is not ordered.

   For more information about these messages, see ctmudly messages (on page 324).

For more information on ctmudly parameters, see ctmudly parameters (on page 322).

ctmudly parameters

The following table lists the ctmudly utility parameters:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td><i><order_ method></i></td><td colspan="2">Name of a order method job that is associated with one or more folders. This parameter is case-sensitive.
■ The specified name must be no longer than 10 characters. If a longer name is specified, it will be truncated to 10 characters.
■ If the ctmudly utility command is issued from a Control-M/Agent, it must include the -DAILY_NAME keyword (as shown above).</td></tr>
<tr><td rowspan="3"><b>-odate</b></td><td colspan="2">Indicates the scheduling date (odate) to be associated with jobs that are ordered by this order method job.
Valid values are:</td></tr>
<tr><td><b>ODAT</b></td><td>The current working date of the computer on which Control-M/Server is running.
This is the default value.</td></tr>
<tr><td><b>yyyymmdd</b></td><td>A specific working day in yyyymmdd format.</td></tr>
</table>

Parameter	Description	
	The interpretation of this parameter value is dependent on the value specified for the -odate_option parameter (described below).	
-odate_option	Indicates how the specified -odate value should be used. Valid values are:	
	value_date	The specified odate should be used as the odate value for the job. However, the jobs ordered by the order method job should be run during the current working day. This is the default value for the -odate_option parameter. If time zones specified in specific job processing definitions in the folders, then the jobs are run according to those time zones.
	run_date	The jobs ordered by the order method job should be run only when the specified odate begins. If the specified odate is the current working day, then this will work in the same way as value_date (described above). If the specified odate has not begun (for example, due to time zone specifications), then the job will wait in the Active Jobs database (with WAIT_ODAT status) until the start of the specified working day.

ctmudly messages

The following table describes the ctmudly utility issued messages:

Message	Description
Security Issues	When a job is not scheduled due to security protection, the following Alert is sent to Control-M/EM: `DAILY <order_ method name>  FAILED TO ORDER JOBNAME <jobName>  - Security`
Scheduling issues	If one or more jobs in a folder is not ordered by a Order method due to scheduling criteria, the type of Alert sent to Control-M/EM is determined by the value of the Control-M configuration parameter NOT_ORDERED_JOB_ALERT. For more information about this parameter, see Configuration parameters.
PARTIAL COPY message	If one or more jobs in a folder was not ordered by a Order method due to scheduling criteria or security settings, the Order method (ctmudly) ended with the following error message in the job output (OUTPUT).

ctmudly examples

The following are examples of the ctmudly utility commands:

ordering a specific Order Method

The following command orders all Folders that are associated with the Order method named prod:

```
ctmudly prod
```

ordering a Order Method for a specific time zone

The following command orders all Folders that are associated with the Japan Order method job, with an odate of March 31, 2008. These jobs will not be run until that working day begins.

```
ctmudly -DAILY_NAME Japan -odate 20080331 -odate_option run_date
```

```
ctmudlst LIST "*"
```

```
ctmudlst UPDATE <order_ method> <date>
```

ordering an Order Method that has a future date specified by using ctmudlst

To specify an ordering date for Order Method named UD_ex1 at a future date, run the following command:

```
ctmudlst update UD_ex1 20090919
```

The following message is displayed:

success in updating the order method 'UD_ex1'

To order UD_ex1, issue the following command:

```
ctmudly UD_ex1
```

The following message is displayed:

Order method: dates mixedup (LASTRUN, new odate)

Affecting the active database Control-M/Server utilities

These utilities, when run, will affect one of the following in the active database. These utilities have an immediate impact on the jobs as they run.

- Prerequisite conditions: the existence or deletion of these conditions can determine if jobs run (ctmcontb)
- Refresh the value of system Control-M/Server parameters by triggering Control-M/Server processes (ctmipc)
- Update the resource usage of a Control-M/Agent (ctmloadset)
- Display and update the contents of a variable (ctmstvar and ctmvar)
- Displays a list of Control resources and the status of each resource, after which the user might decide to modify the status of a resource (ecactltb)

By including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utility	Description
ctmcontb (on page 327)	Performs operations on the Prerequisite Conditions folder.
ctmipc (on page 332)	Refreshes recently updated parameters, as an alternative to restarting Control-M/Server.
ctmloadset (on page 334)	Updates a resource in the Quantitative Resources folder with regard to usage on an agent computer.
ctmstvar (on page 337)	Displays the current value of a variable.
ctmvar (on page 339)	Manipulates Global variables for data centers, SMART Folders (excluding Sub-folders), or jobs in SMART Folders.
ecactltb (on page 343)	Lists the status of each resource in the Control Resources table.

ctmcontb

The ctmcontb utility performs the following operations on prerequisite conditions in the Control-M/Server database:

- Lists existing prerequisite conditions while treating an asterisk in a condition name as a wildcard. (-LIST)
- Lists existing prerequisite conditions while treating an asterisk in a condition name as a regular character. (-LISTNOWILD)
- Adds a prerequisite condition. (-ADD)
- Generates a prerequisite condition in XML format. (-XML)
- Deletes a prerequisite condition while treating an asterisk in a condition name as a wildcard. (-DELETE)
- Deletes a prerequisite condition while treating an asterisk in a condition name as a regular character. (-DELETENOWILD)
- Deletes conditions within a specified range of dates. (-DELETEFROM)

  The following special characters are disabled when they occur in prerequisite condition names:

  - (open parenthesis
  -) close parenthesis
  - | vertical bar
  - space

To run the utility, see Running the ctmcontb utility (on page 327).

Running the ctmcontb utility

This procedure describes how to run the ctmcontb utility, which enables you to perform operations on prerequisite conditions in the Control-M/Server database.

➢ To run the ctmcontb utility:

- Do one of the following:
  - Run one of the following commands to list conditions:

    -
      ```
      ctmcontb -LIST <conditionName> <conditionDate>
      [-output <outputFileName>][-FULLDETAILS]
      ```

    -
      ```
      -ctmcontb -LISTNOWILD <conditionName> <conditionDate>
      [-output <outputFileName>][-FULLDETAILS]
      ```

    -
      ```
      ctmcontb -input_file <fullPathFileName>
      ```
  - Run one of the following commands to delete a range of conditions:

    -
      ```
      ctmcontb -DELETEFROM <conditionName> <fromDate> <toDate>
      ```

    -
      ```
      -ctmcontb -input_file <fullPathFileName>
      ```
  - Run one of the following commands for all other operations:

- o `ctmcontb {-ADD|-XML|-DELETE|-DELETEONWILD} <conditionName> <conditionDate>`
- o `ctmcontb -input_file <fullPathFileName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmcontb parameters, see ctmcontb utility parameters (on page 329).

ctmcontb utility parameters

The following table lists the ctmcontb utility parameters:

Variable	Description
-FULLDETAILS	Displays the prerequisite condition name without truncation.
<conditionName>	**For LIST, DELETE and DELETEFROM** The condition name can include wildcard character * to indicate any number of characters (including none). When using an *, the condition name must be enclosed in quotation marks (for example, "LVL*"). Specify "*" by itself to include all existing conditions. When using both open and closed square brackets ([and]), the condition name must be enclosed in quotation marks (for example, "RATE[A1]"). Maximum length for a condition name is 255 characters.
<conditionName> continued	**For LISTNOWILD and DELETENOWILD** The character * in condition name is referred to as a regular character. When using an *, the condition name must be enclosed in quotation marks (for example, "LVL*"). When using both open and closed square brackets ([and]), the condition name must be enclosed in quotation marks (for example, "RATE[A1]"). Maximum length for a condition name is 255 characters. **For Adding or Generating in XML Format** When using both open and closed square brackets ([and]), the condition name must be enclosed in quotation marks (for example, "RATE[A1]"). Maximum length for a condition name is 255 characters

Variable	Description
<conditionDate>	Displays the date of the prerequisite condition in mmdd format. **For Listing and Deleting** The condition date can include wildcard character * to indicate any number of characters (including none). When using an *, enclose the date in quotation marks (for example, "12* "). ▪ Specify "*" by itself to include all dates. ▪ Specify ODAT to accept the Control-M date. ▪ Specify STAT if a date is not relevant. **For Adding or Generating in XML Format** ▪ Specify ODAT to use the Control-M working date. ▪ Specify STAT if a date is not relevant.
<fromDate> <toDate>	**For Deleting in a Date Range** Displays the starting and ending dates for date range of prerequisite conditions to delete. Each date is in *mmdd* format. If the To Date is less than the From Date, the range of condition dates will include the From Date up to the end of the year (1231) plus the beginning of the next year (0101) up to the To Date.
<outputFileName>	**For Listing** Displays the full path name to which the report should be sent (optional). If this parameter is not specified, the output is routed to the default output device (the terminal).
<fullPathFileName>	Displays the name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: prepare, save, and reuse files of utility parameters. specify utility input longer than the number of characters allowed in the command line. `-input_file /ctm_server/data/ctmcontb_list.txt`

ctmcontb examples

The following are examples of the ctmcontb utility commands:

The following command deletes prerequisite condition bckp_end with condition dates in December:

```
ctmcontb -DELETE bckp_end "12*"
```

The following command deletes all prerequisite conditions with prefix a and condition dates between December 1 and December 15 inclusive:

```
ctmcontb -DELETEFROM "a*" 1201 1215
```

The following command deletes the prerequisite condition aa* with condition date ODAT:

```
ctmcontb -DELETENOWILD "aa*" ODAT
```

You can implement the second example with the -input_file parameter as follows:

```
ctmcontb -input_file
~<controlm_owner>/ctm_server/data/ctmcontb_delfr.txt
```

Where the referenced file contains the line:

```
-DELETEFROM "a*" 1201 1215
```

The following command lists the prerequisite condition named aa* with all its dates:

```
Ctmcontb -LISTNOWILD "aa*" "*"
```

The following command lists all existing prerequisite conditions:

```
ctmcontb -LIST "*" "*"
```

A report similar to the following example is generated:

```
Date: 30-JUN-2008. Page 1

Conditions list

CONDNAME     CONDDATE

APR01-L20    0629

APR01-L20    0630

ARD01-L30K   0630

LVL11-LVL22  0628

LVL11-LVL22  0629

LVL11-LVL22  0630

PKR11-LVL01  0630
```

This example demonstrates the advantage of defining a Control-M job to run a utility. The following job processing definition causes Control-M to run ctmcontb each work day, each time deleting all prerequisite conditions that are between five and ten days old:

```
Week Days 2,3,4,5,6

Variable Assignment
%%A=%%CALCDATE %%DATE -10

%%B=%%CALCDATE %%DATE -5
```

```
%%A=%%SUBSTR %%A 3 4
%%B=%%SUBSTR %%B 3 4
Command Line ctmcontb -DELETEFROM "*" %%A %%B
```

This example illustrates ctmcontb input and output when using the -XML option. ODAT automatically generates the Control-M/Server system date that, in this example, was March 15th.

```
D:\>ctmcontb -XML cond1 ODAT
<?xml version="1.0" encoding="utf-8" ?>
<CTMCONTB
 CONDNAME="cond1"
 CONDDATE="0315">
 <COND
CONDNAME="cond1"
CONDDATE="    0315">
  </COND>
</CTMCONTB>
```

ctmipc

The ctmipc utility sends a message to Control-M/Server processes, which instructs the processes to perform a specific action. To run the ctmipc utility, see Running the ctmipc utility (on page 332).

Currently, the ctmipc utility can only be used to refresh recently updated parameters, as an alternative to restarting Control-M/Server. For more information, see the example below.

Run this utility according to your operating system, as follows:

- On Windows: At the command prompt
- On UNIX: From the Control-M/Server connection profile

Specify the following command to send the message "CFG" to all Control-M/Server processes:

```
ctmipc -dest ALL -msgid CFG
```

Running the ctmipc utility

This procedure describes how to run the ctmipc utility, which enables you to send a message to Control-M/Server processes, which instructs the processes to perform a specific action.

➢ To run the ctmipc utility:

- Type the following command:

```
ctmipc
  -DEST    <CO|CD|CS|SL|SU|LG|NS|TR|WD|RT|AC|ALL>
  -MSGID   <CFG>
```

```
    { -DATA  <message data>]
    [ -QUIET ]
    [ -DEBUG  <debug level> ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmipc parameters, see ctmipc parameters (on page 333).

ctmipc parameters

The following table lists the parameters for the ctmipc utility:

Parameter	Description
-DEST	Indicates the destination of the message to be sent (either one specific process, or specify .DEST ALL for all processes).
-MSGID	Indicates the message to be sent.
-DATA	Indicates additional data to be sent with the message ID (if any).
-QUIET	Indicates that the ctmipc utility will not generate any output message, upon success or a failure.
-DEBUG	Indicates that the utility will run in debug mode, and will therefore generate a debug log in the proclog sub-directory of Control-M/Server.

ctmloadset

The ctmloadset utility records current resource usage on an agent computer in the Quantitative Resources table. This utility is typically invoked by a cyclic job that runs on the agent computer and measures usage of a certain resource on the computer. Usage data is then used to update the Quantitative Resources table on the Control-M/Server computer. To run the ctmloadset utility, see Running the ctmloadset utility (on page 335).

ctmloadset is used when load balancing is implemented. The load-balancing algorithm uses the data recorded in the Quantitative Resources table to determine to which agent computer a job should be submitted.

Control-M maintains the following information about usage of each Quantitative resource:

- **Total Used**: Units of the resource currently in use. This parameter represents the sum of the values specified in the other two rows of this table.
- **Used by Control-M**: Units of the resource currently in use by jobs submitted by Control-M for Databases.
- **Used by Others**: Units of the resource currently in use by non-Control-M jobs.

Update resource usage values in the Quantitative Resources table in either of two ways:

- Specify the value for Total Used for a resource. ctmloadset subtracts the value for Used by Control-M from the value you specify and places the remainder in the field Used by Others.
- Specify the value for Used by Others for a resource. This value is added to the value Used by Control-M to calculate the value Total Used for the resource.

Values for the utility can be expressed as an absolute number of units or as a percentage of the total number of units defined (Max value).

The utility's output is displayed as type Q rows in the Quantitative Resources window. However, the Mem Name field remains blank since this represents usage by one or more non-Control-M jobs.

A host group contains three agent computers: diana, jacklin and ruby. Each computer is defined in the Quantitative Resource table as having 200 units of resource CPU_load, representing the load on the computer's CPU.

- computer jacklin is used exclusively to run jobs submitted by Control-M. The computer is currently executing a job that uses 120 units of resource CPU_load.
- computer ruby is used exclusively to run jobs submitted by Control-M. The computer is currently executing a job that uses 150 units of resource CPU_load.
- computer diana is used both for Control-M and non-Control-M jobs. The computer is currently executing a job submitted by Control-M that uses 75 units of resource CPU_load.

A cyclic job is defined to run periodically on diana to measure the total load on the CPU. The job updates the Quantitative Resources table using the ctmloadset utility to indicate to Control-M exactly what the load is on that computer. The last run of this job determined that the load on the CPU is 80% of total capacity. The job invokes ctmloadset as follows:

```
ctmloadset TOTAL CPU@diana 80%
```

The Total Used for diana is set to 80% of 200, or 160. Since the usage by Control-M jobs is currently 75 units, ctmloadset calculates that the "Other" (non-Control-M usage) is 160 \-\-— 75, or 85.

As a result, the Quantitative Resources table now contains the following values:

Resource		Total used by Control-M	Total used by others	
CPU@jacklin		120		
CPU@ruby		150		
CPU@diana		75	85	

The Control-M load-balancing algorithm uses these values when determining where to submit the next job.

Running the ctmloadset utility

This procedure describes how to run the ctmloadset utility, which enables you to record current resource usage on an agent computer in the Quantitative Resources table.

➢ To run the ctmloadset utility:

- Specify the following command to invoke the ctmloadset utility:

```
ctmloadset {TOTAL|OTHERS} <QRname> <loadValue>[%]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmloadset parameters, see ctmloadset parameters (on page 335).

ctmloadset parameters

The following utility lists the ctmloadset parameters:

Parameter	Description
TOTAL	Indicates that the load value provided specifies the total usage of the resource by all jobs (both Control-M jobs and non-Control-M jobs). When this option is specified, the utility calculates the usage of the resource by non-Control-M jobs and updates the table accordingly.

Parameter	Description
OTHERS	Indicates that the load value provided specifies the units of the resource used by one or more non-Control-M jobs.
<QRname>	Name of the Quantitative resource to update.
<loadValue>	Number of units of the resource currently used. *-or-* When % is specified, the amount of the resource currently used, expressed as a percentage of the maximum available units defined for this Quantitative resource.

ctmloadset examples

The following examples demonstrate the effect of ctmloadset on the Quantitative Resources table, as represented by the display generated by the ecaqrtab utility. All examples below are based on the following premise:

For agent computer diana, 30 units of resource CPU@diana are currently used by Control-M jobs.

The output from the ecaqrtab utility is as follows:

Resource Name		Max-Avail	Reserved		
CPU@diana		50	0		

The following command specifies that the current usage of the Quantitative resource CPU@diana by non-Control-M jobs is 12 units:

```
ctmloadset OTHERS CPU@diana 12
```

As a result, the output from the ecaqrtab utility is now as follows:

Resource Name		Max-Avail	Reserved		
CPU@diana		50	0		

The following command specifies that the current usage of the Quantitative resource CPU@diana by non-Control-M jobs is 12%:

```
ctmloadset OTHERS CPU@diana 12%
```

The non-Control-M usage of the resource is calculated as 12% of 50, or 6 units. As a result, the output from the ecaqrtab utility is now as follows:

Resource Name		Max-Avail	Reserved		
CPU@diana		50	0		

The following command specifies that the current total usage of the Quantitative resource CPU@diana by all jobs is 48 units:

```
ctmloadset TOTAL CPU@diana 48
```

As a result, the output from the ecaqrtab utility is now as follows:

Resource Name		Max-Avail	Reserved		
CPU@diana		50	0		

ctmstvar

The ctmstvar utility displays the current value of a variable or function. To run the ctmstvar utility, see Running the ctmstvar utility (on page 338).

- **UNIX only**:
  - A variable that does not contain a $ sign can be enclosed in single or double quotation marks.
  - A variable that does contain a $ sign must be enclosed in single quotation marks.
  - A variable that contains a $ sign cannot be resolved if it is enclosed in double quotation marks.
- **Windows only***:* Variables must be enclosed with double quotation marks.

Running the ctmstvar utility

This procedure describes how to run the ctmstvar utility, which enables you to display the current value of a variable or function.

➢ To run the ctmstvar utility:

- Type the following command to invoke the ctmstvar utility:

```
ctmstvar <orderID> <variableString>
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmstvar parameters, see ctmstvar parameters (on page 338).

ctmstvar parameters

The following table lists the ctmstvar utility parameters:

Variable	Description
<orderID>	Order ID of a job waiting in the Active Jobs database (as displayed in the Job Details window of Control-M/EM). The Order ID displayed in Control-M/EM is a base 36 number. If you want to specify the Order ID here as a base 10 number, prefix the number with an asterisk, and enclose it in quotation marks (for example,"*1234"). Use "0" to indicate no specific Order ID.
<variableString>	The variable or string enclosed in quotation.

ctmstvar examples

The following are examples of the ctmstvar utility commands:

For UNIX

```
ctmstvar a1 '%%$CALCDATE %%ODATE -2'
ctmstvar 0  "%%ODATE"
```

For Windows

```
ctmstvar a1 "%%$CALCDATE %%ODATE -2"
ctmstvar 0  "%%ODATE"
```

ctmvar

The ctmvar utility defines, deletes, modifies and displays variables. This utility can be applied to variables that are:

- In a specific job processing definition in a SMART Folder
- Common to all jobs in a SMART Folder
- Global for an entire data center (a Control-M for Databases and all associated agents)

To run the ctmvar utility, see running the Running the ctmvar utility (on page 339).

Consider the following:

- If a SMART Folder specified in the ctmvar utility has been ordered more than once, the utility updates every instance of that SMART Folder in the Active Jobs database.
- Variables in jobs that are not part of a SMART Folder cannot be modified using the ctmvar utility.
- A value specified for a Global variable is overridden if a local variable with the same name is defined in a job processing definition, Sub-folder or SMART Folder.
- Sub-folders cannot be modified using the ctmvar utility.

For more information about variables, see Control-M Variable facility .

Running the ctmvar utility

This procedure describes how to run the ctmvar utility, which enables you to define, delete, modify and display variables.

➢ To run the ctmvar utility:

- Specify one of the following commands:
  - The ctmvar utility must always start with %%

```
ctmvar
  -ACTION                 <LOAD|SET|DELETE|LIST>
  [ -VAR                    <variable> ]
  [ -VAREXPR                <variableExpression> ]
  [ -FILENAME               <filename> ]
  [ -QUIET ]
  [ -DEBUG                  <debug level> ]
```

  - `ctmvar -input_file <fullPathFileName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmvar parameters, see ctmvar parameters (on page 340).

ctmvar parameters

The following table lists the ctmvar utility parameters:

Parameter	Description	
-ACTION	Indicates the action to be performed on the specified variable. The possible actions are:	
	LOAD	Loads variables from a file. When this option is used, parameter -filename is required. The format for each variable in the specified file is: %%[\<SMARTtble>[\<job>]]\<varName>=<expression> If the variable does not exist in the data center or the specified SMART Folder or job, it is created. If the variable already exists, it is updated with the specified value.
	SET	Defines a new variable. When this option is used, parameters -var and -varexpr are required. If the variable does not exist in the data center or the specified SMART Folder or job, it is created. If the variable already exists, it is updated with the specified value.
	DELETE	Deletes a Global variable. When this option is used, the -var parameter is mandatory. This action cannot be used for variables that have been defined for a specific job or SMART Folder.

Parameter	Description	

	LIST	Displays all Global variables for the data center or all variables for the specified SMART Folder specified in the -var parameter.
		`ctmvar -action LIST`
		Displays all Global variables for the data center.
		`ctmvar -action LIST -var "%%\PAYROLL"`
		Displays all variables that are global for the PAYROLL SMART Folder.
		Variable values can also be displayed using the ctmstvar utility. However, the ctmstvar utility resolves the current value of only a specified variable or function. The ctmvar utility displays all variables in the data center or the specified SMART Folder.

-VAR	Name and location of the variable that the specified action should be applied to. The valid format for this parameter depends on the type of variable being handled. For a variable that is global for an entire data center, valid format is: -var "%%\<var_name>" For a variable that is global for all jobs in a SMART Folder, valid format is: -var "%%\<SMART_folder_name>\<var_name>" For variable in a specific job in a SMART Folder, valid format is: -var "%%\<folder_name>\<jobName>\<var_name>" This parameter cannot be specified together with -action LOAD. For more information about variables, see Control-M Variable facility .	

Parameter	Description
-VAREXPR	Value to be assigned to the specified variable. The specified value can be: a string (embedded in quotation) an integer (a numeric value) an Variable expression (for example, with an Variable function) another (existing) global variable. This parameter cannot be specified together with -action LOAD. For more information, see Control-M Variable facility .
-FILENAME	Path and name of the file containing the list of variables. The file must be accessible to Control-M/Server. This parameter is only valid when specified together with -action LOAD. The syntax for each line in the specified file is %%[\<*ctmvar*>[\<*job*>]]\<*varName*>=<*expression*> Specify the entire pathname in this parameter.
-QUIET	Suppresses the display of the results.
-DEBUG	Sets a debug level for the utility. This parameter is used for maintenance and troubleshooting purposes. The level, a numeric value from 0 to 5, must be used only if requested and specified by Technical Support. Performance is somewhat slower and requires a larger number of resources when operating in debug mode. BMC recommends that you activate debug mode only when absolutely necessary and revert to normal mode as soon as possible.
-input_file	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: ▪ prepare and save files of utility parameters that can be reused. ▪ specify utility input longer than the number of characters allowed in the command line. -input_file ~<controlm_owner>/ctm_server/data/ctmvar_parms.txt

ctmvar examples

The following are examples of the ctmvar utility:

The following command assigns the value UP to variable %%CTMSTATUS:

```
ctmvar -action set -var "%%\CTMSTATUS" -varexpr "UP"
```

The following command assigns the value 31 to variable %%MONTHDAYS in the folder called PAYROLL:

```
ctmvar -action set -var "%%\PAYROLL\MONTHDAYS"
       -varexpr 31
```

The following command assigns the current value of system variable %%TIME to variable %%AAA:

```
ctmvar -action set -var "%%\AAA" -varexpr %%TIME
```

You can get the same result using the -input_file parameter as follows:

```
ctmvar -input_file
~<controlm_owner>/ctm_server/data/var_expr_parms.txt
```

The referenced file contains the following lines:

- `-action set`
- `-var "%%\AAA"`
- `-varexpr %%TIME`

The format variable %%@varname indicates that the variable should contain a value to be resolved by each job that uses it. In the following example, the command assigns the value %%@TIME to variable %%AAA:

```
ctmvar -action set -var "%%\AAA" -varexpr %%@TIME
```

ecactltb

The ecactltb utility displays a list of Control resources and the status of each resource. To run the ecactltb utility, see Running the ecactltb utility (on page 343).

If the name of an output file is specified, but no path is specified for this file, the output file is placed in the Control-M/Server home directory.

Running the ecactltb utility

This procedure describes how to run the ecactltb utility, which enables you to display a list of Control resources and the status of each resource.

➢ To run the ecactltb utility:

- Type the following command:

  ```
  ecactltb [<output>]
  ```

  <*output*> is the full path name to which the report should be sent (optional). If this parameter is not specified, the output is routed to the default output device.

The following command generates a list of Control resources in the file rprt.txt.

```
ecactltb ~<controlm_owner>/ctm_server/user1/rprt.txt
```

Communication, startup, and troubleshooting

The communication, startup, and troubleshooting utilities are used to set up communication between Control-M components, start up/shut down Control-M components and entities, and determine if communication between the components is occurring effectively.

Various troubleshooting utilities are also included in this chapter.

By including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utility	Description
ag_diag_comm (on page 401)	Verifies communication between Control-M/Agent and Control-M/Server.
Running the ag_ping utility (on page 406)	Verifies that Control-M/Server is active on computer connected to a Control-M/Agent.
ctl (on page 347)	Checks if Control-M/EM Server components are operational and send commands.
ctm_agstat (on page 372)	Lists or updates the status of a Control-M/Agent.
ctm_diag_comm (on page 374)	Generates a report on the connection details of a Control-M/Agent and its remote host computer.
ctmgetcm (on page 376)	Collects and displays Control-M/Agent application information.
ctmhostmap (on page 378)	Manages the mapping of remote hosts to Control-M/Agents and the conversion of Control-M/Agents to remote host computers.
ctmhostgrp (on page 384)	Enables the maintenance and viewing of host groups.
ctmping (on page 386)	Collects configuration information about Control-M/Agents and test communications.

Utility	Description
ctmshout (on page 390)	Issues a Shout message to a specified destination.
ctmshtb (on page 393)	Sets the active Shout Destination folder.
ctmspdiag (on page 393)	Prints or erases diagnostics from stored procedures and set or show diagnostic request status of stored procedures.
ctmsuspend (on page 396)	Suspends Control-M/Server scheduling processes for mass uploads/downloads from Control M/EM.
init_prflag (on page 397)	Resets sleep times and trace levels for Control-M/Server processes.
orbadmin (on page 365)	Manages the Naming Service process and the CORBA configuration file.
Running the shagent utility (on page 406)	Shows if an agent and tracker are running.
Running the shut_ca utility (on page 399)	Shuts down the Control-M/Server Configuration Agent.
Running the shut_ctm utility (on page 399)	Shuts down Control-M/Server and its processes.
Running the shutdb utility (on page 400)	Shuts down the SQL database server.
Running the start_ca utility (on page 400)	Starts up the Control-M/Server Configuration Agent.
Running the start_ctm utility (on page 400)	Starts up Control-M/Server.
Running the startdb utility (on page 401)	Starts up the SQL database server.

ctl

The ctl command line utility enables you to send simple requests to Control-M/EM server components. The ctl utility can:

- Check if Control-M/EM server components are operational.
- Check if Batch Impact Manager and Control-M/Forecast (Forecast) server components are operational (if they are installed at your site).
- Send commands to Control-M/EM components during runtime, for example, setting Gateway debug level.

The ctl utility is automatically installed with a full Control-M/EM installation.

To run the utility, you can select one of the following:

- Running the ctl utility on Windows (on page 348)
- Running the ctl utility on UNIX (on page 348)
- Running the ctl utility from the Control Shell (on page 348)

The command syntax for the ctl utility is a function of the component for which the utility runs. The parameters and syntax of the ctl utility are described in the following:

- Parameters common to all ctl commands (on page 349)
- Accessing the Global Conditions server from the ctl utility (on page 350)
- ctl-Global Conditions server parameters (on page 350)
- Accessing the GUI server from the ctl utility (on page 351)
- ctl-GUI server parameters (on page 352)
- Accessing Gateway from the ctl utility (on page 353)
- ctl- Gateway parameters (on page 354)
- Accessing the Forecast server from the ctl utility (on page 356)
- ctl-Forecast server parameters (on page 358)
- Accessing the Configuration Management server from the ctl utility (on page 359)
- ctl-Configuration Management server parameters (on page 360)
- Accessing the Configuration Agent from the ctl utility (on page 361)
- ctl-Configuration Agent parameters (on page 361)
- Acessing the BMC Batch Impact Manager server from the ctl utility (on page 362)
- ctl-BMC Batch Impact Manager server parameters (on page 363)
- Accessing the Control-M Self Service server from the ctl utility (on page 363)
- ctl- Self Service server parameters (on page 364)

Running the ctl utility on Windows

This procedure describes how to run the ctl utility on Windows.

➢ To run the ctl utility on Windows:

1. Open a command prompt window.
2. Specify the ctl command line.

Running the ctl utility on UNIX

This procedure describes how to run the the ctl utility on Unix.

➢ To run the ctl utility on UNIX:

1. Open an Xterm window.
2. Log on as the Control-M/EM installation account user.
3. Run the em ctl command from the command line as the Control-M/EM database owner (DBO).

Running the ctl utility from the Control Shell

This procedure describe how to run the ctl utility from teh Control Shell, which enables you to to send commands to the Control-M/EM server components. These commands are the same as those that you can run under the -cmdstr parameter, in the ctl command line utility.

➢ To perform commands using the Control Shell:

1. Right-click the required Control-M/EM server component, and select **Control Shell**.

   The Control Shell dialog box is displayed.
2. Click **Usage** to populate the **Result** field of the Control Shell dialog box with the commands and requests available for the specific component.
3. Select the required command, and copy it into the **Specify...** field of the Control Shell dialog box. You can also manually enter a specific command.
4. Click **Apply**.

Parameters common to all ctl commands

The following table lists the parameters that are present in the syntax of all variations of the ctl command. The parameters specific to each server component are discussed separately in the relevant sections, as follows:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>em</td><td colspan="2">Prefix to be specified when running this utility on a UNIX operating system.</td></tr>
<tr><td>-U</td><td colspan="2">Control-M/EM database user name.</td></tr>
<tr><td>-P</td><td colspan="2">Control-M/EM database user password.</td></tr>
<tr><td>-pf</td><td colspan="2">Flat file containing an unencrypted username and password on separate lines in the format:
user=<i>username</i>
password=<i>password</i></td></tr>
<tr><td>-reg</td><td colspan="2">Checks if the Global Conditions Server is registered in the CommReg table. -reg cannot be used with -cmd or -cmdstr.</td></tr>
<tr><td rowspan="3">-cmd</td><td colspan="2">Indicates a command to be performed by the Global Conditions Server.
-cmd cannot be used with -reg.</td></tr>
<tr><td>stop</td><td>Stops the Global Conditions Server.
This command cannot be specified with other commands in the same run of the ctl utility.</td></tr>
<tr><td>life_check</td><td>Checks if the Global Conditions Server is active.
This command cannot be specified with other commands in the same run of the ctl utility.</td></tr>
<tr><td>-timeout</td><td colspan="2">Indicates the period of time (in seconds) that ctl waits for a response from the GUI Server before declaring that communication has failed.
Default: 30. Optional.</td></tr>
<tr><td>-diagon</td><td colspan="2">Activates tracing of ctl work flow (diagnostics). The results are written to the <b>ctl_diag.machine.txt</b> file located in the working directory. Optional.</td></tr>
</table>

Accessing the Global Conditions server from the ctl utility

This procedure describes how to acccess the Global Conditions server from the ctl utility, which enables you to send simple requests to the Global Condtions server.

➢ To access the Global Conditions server:

- Type the following command:

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
  -C GCS {-M <computerName> | -all}
{ -reg |
  -cmd stop |
  -cmd life_check |
  -cmd change_file |
  -cmdstr "<commandString>"}]
 [-timeout <<responseTimeout> (seconds)>]
 [-diagon]
```

For more information, see ctl-Global Conditions server parameters (on page 350).

ctl-Global Conditions server parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the Global Conditions Server are described in the following table:

Parameter	Description
-C GCS	Global Conditions Server to which the command is directed. The Global Conditions Server handles the distribution of conditions that affect jobs in more than one data center.
-M	Specifies a computer name. This name is used to identify the computer to which the Global Conditions Server belongs. -M cannot be used with -all.
-all	Directs a query or command to all Global Conditions Servers. -all cannot be used with -M.
-cmd	Indicates a command to be performed by the Global Conditions Server. -cmd cannot be used with -reg.
	The following -cmd value is used for diagnostic and debugging purposes, which is described in Control-M diagnostics. Diagnostic and non-diagnostic commands can be specified on the same ctl command line.

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td></td><td>change_file</td><td>Closes the current log and diag files and creates new files.</td></tr>
<tr><td>-cmdstr</td><td colspan="2">Specifies a text string to be sent to the Global Conditions Server. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" ").
-cmdstr cannot be used with -reg or -cmd.
The -cmdstr parameter is used for diagnostic purposes only. For more information about the -cmdstr parameter, see System configuration.</td></tr>
</table>

Accessing the GUI server from the ctl utility

This procedure describes how to access the GUI server from the ctl utility, which enables you to send simple requests to the GUI server.

➢ To access the GUI server:

- Type the following command:

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
 -C GUI_Server
 {-M <computerName> | -name <logicalName> | -all}
 {-reg |
  -cmd stop |
  -cmd life_check |
  -cmd do_measure |
  -cmd get_measure |
  -cmdstr "<commandString>"}
 [-timeout <<responseTimeout> (seconds)>]
 [-diagon]
```

To manually refresh version management system parameters without restarting the GUI server, issue the following command:

```
ctl -u <user> -p <password> -C GUI_Server -name <GSRName> -cmdstr
REFRESH_HISTORY
```

For more information, see ctl-GUI server parameters (on page 352).

ctl-GUI server parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the GUI server are described in the following table:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-C GUI_Server</td><td colspan="2">Indicates that the command is directed to one or more GUI servers. GUI servers handle communications between Control-M for Databases GUI workstations and other Control-M for Databases components.</td></tr>
<tr><td>-M</td><td colspan="2">Specifies a computer name. This name is used to identify the computer to which the GUI server belongs. -M cannot be used with -all. When -M is specified, the request is sent to the GUI server whose name is equal to the value indicated by -M.</td></tr>
<tr><td>-name</td><td colspan="2">Logical name of the GUI server. If the GUI server is started without specifying -name, the logical name of the GUI server is equal to the host name of the computer where the GUI server is running.</td></tr>
<tr><td>-all</td><td colspan="2">Directs a query or command to all GUI servers. -all cannot be used with -M.</td></tr>
<tr><td rowspan="3">-cmd</td><td colspan="2">Indicates a command to be performed by the GUI server. -cmd cannot be used with -reg.</td></tr>
<tr><td>do_measure</td><td>Initiates collection of statistics about the GUI server.
This command cannot be specified with other commands in the same run of the ctl utility.</td></tr>
<tr><td>get_measure</td><td>Retrieves statistics from the GUI server and displays them.
This command cannot be specified with other commands in the same run of the ctl utility.</td></tr>
<tr><td>-cmdstr</td><td colspan="2">Specifies a text string to be sent to the GUI server. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" ").
-cmdstr cannot be used with -reg or -cmd.
Valid values include:
REFRESH_LDAP
REFRESH_HISTORY
For more information about the -cmdstr parameter, see System configuration.</td></tr>
</table>

Accessing Gateway from the ctl utility

This procedure describes how to access Gateway from the ctl utility, which enables you to send simple requests to the Gateway.

➢ To access Gateway from ctl utility:

- Type the following command:

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
-C Gateway {-dc <dataCenter> |-all}
{-reg |
-cmd stop |
-cmd life_check |
{[-cmd dwl]
 [-cmd no_dbg]
[-cmd db{0-9}]
 [-cmd gui{+|-}]
[-cmd host{+|-}]
[-cmd trunc{+|-}]
 [-cmd alive{+|-}]
[-cmd job{+|-}]
 [-cmd dwl_debug{+|-}]
[-cmd hostlink{+|-}]
 [-cmd guilink{+|-}]
 [-cmd show_jcl]}}
[-cmdstr "<commandString>"]
 [-timeout <<responseTimeout> (seconds)>]
[-diagon]
```

For more information, see ctl- Gateway parameters (on page 354).

ctl- Gateway parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the Gateway are described in the following table:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>RETRIVE_USERDAILYORDER_METHOD</td><td colspan="2">Directs a request to Control-M for z/OS for updated Order method information to be sent to the Control-M/EM database.</td></tr>
<tr><td>-C Gateway</td><td colspan="2">Indicates that the command is directed to the gateway. The gateway mediates between Control-M for Databases components and the Control-M installation.</td></tr>
<tr><td>-dc</td><td colspan="2">Name of a data center. This name identifies the gateway to which ctl is sending a command or message. This parameter is used when a query or command is directed to a specific gateway (as specified by the -C parameter). -dc cannot be used with -all.</td></tr>
<tr><td>-all</td><td colspan="2">Directs a query or command to all components of the gateway (as specified by the -C parameter).
-all cannot be used with -dc.</td></tr>
<tr><td rowspan="8">-cmd</td><td colspan="2">Indicates a command to be performed by the gateway.
-cmd cannot be used with -reg.</td></tr>
<tr><td><b>dwl</b></td><td>Forces a new download from a Control-M installation.</td></tr>
<tr><td colspan="2">The following -cmd values are used for diagnostics and debugging, as described in Control-M diagnostics.
Diagnostic and non-diagnostic commands can be specified on the same ctl command line.</td></tr>
<tr><td>change_file</td><td>Closes the current log and diag files and creates new files.</td></tr>
<tr><td>DIAG [on|off]</td><td>Start and stop DIAG. Valid values:
■ on – enable diag facility
■ off – disable diag facility</td></tr>
<tr><td>no_dbg</td><td>Stops all debug printing.</td></tr>
<tr><td>db#</td><td>Debug level for database operations. Range: 0 - 9
0 turns off debugging.</td></tr>
<tr><td>gui +| –</td><td>Starts or stops a debug trace for the GUI.</td></tr>
<tr><td></td><td>host +| –</td><td>Starts or stops output of host debug messages.</td></tr>
</table>

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td></td><td>trunc +| –</td><td>Starts or stops truncating messages. Only the message header and one row of data remain after truncation.</td></tr>
<tr><td></td><td>alive +| –</td><td>Starts or stops debugging of "keep alive" messages.</td></tr>
<tr><td></td><td>job +| –</td><td>Starts or stops dumping job messages. During a debug trace, <i>job+</i> displays messages on the screen about active job downloads, active job updates, and Folder uploads. This option can also be set by specifying the trace_job_message [on | off] parameter at the command prompt.</td></tr>
<tr><td></td><td>dwl_debug +| –</td><td>Starts or stops a debug trace for the download procedure.</td></tr>
<tr><td></td><td>hostlink +| –</td><td>Starts or stops a debug trace for the host link.</td></tr>
<tr><td></td><td>guilink +| –</td><td>Starts or stops a debug trace for the GUI link.</td></tr>
<tr><td></td><td>show_jcl</td><td>Shows active JCL.</td></tr>
<tr><td>-cmdstr</td><td colspan="2">Specifies a text string to be sent to the gateway. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" ").
-cmdstr cannot be used with -reg or -cmd.
Valid values include:</td></tr>
<tr><td></td><td>DIAG [on | off]</td><td>(either enable or disable the diagnostic facility)</td></tr>
<tr><td></td><td>DIAGL &lt;context> <level>
[<buffer level&gt;]</td><td>(sets the diagnostic level)</td></tr>
<tr><td></td><td>DIAGSTACKS [on | print]</td><td>(enables the diagnostic stacks option, or prints the details of the current stacks)</td></tr>
<tr><td></td><td>TRACE_CLIENT <on|off&gt;</td><td>(set GTW-Clients trace either on or off)</td></tr>
<tr><td></td><td>TRACE_CTM <on|off&gt;</td><td>(set GTW-CTM trace either on or off)</td></tr>
<tr><td></td><td>TRACE_DB
&lt;0 - 8&gt;</td><td>(set database trace level)</td></tr>
<tr><td></td><td>TRACE_KLIVE <on|off&gt;</td><td>(set keep alive trace either on or off)</td></tr>
</table>

Parameter	Description	
	TRACE_LINK_ CLIENT <on|off>	(set GTW-Clients link level trace either on or off)
	TRACE_LINK_ CTM <on|off>	(set GTW-CTM link level trace either on or off)
	TRACE_DISABLE_ ALL	(turns off trace options when selected)
	TRACE_SNMP <on|off>	(set trace for SNMP traps either on or off)
	TRACE_TRUNC <on|off>	(set truncate trace messages either on or off)
	TRACE_TRUNC_INFO <on|off>	(set truncate trace information messages either on or off)
	TRACE_JOB_ MESSAGE <on|off>	(set job message trace either on or off)
	TRACE_LEVEL_ REPORT	(displays the status of the trace)
	The -cmdstr parameter is used for diagnostic purposes only. For more information about the -cmdstr parameter, see System configuration. The ctl utility can be run from the Control Shell. For more information, see To perform commands using the Control Shell section in ctl (on page 347).	

Accessing the Forecast server from the ctl utility

This procedure describes how to access the Forecast server from the ctl utility, which enables you to send simple requests to the Forecast server.

➢ To access the Forecast server:

- Type the following command

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
  -C <forecastServer>
  {-M <computerName>
  -name <logicalName> | -all}
```

```
    {-reg |
     -cmd stop |
     -cmd life_check |
     -cmdstr "<Command_String>"}
    [-timeout <<responseTimeout> (seconds)>]
    [-diagon]
```

For more information, see ctl-Forecast server parameters (on page 358).

ctl-Forecast server parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the Forecast server are described in the following table:

Parameter	Description
-C forecast Server	Indicates that the command is directed to a Forecast server (if Control-M/Forecast is installed at your site).
-M	Specifies a computer name. This name is used to identify the computer to which the Forecast server belongs.
-name	Logical name of the Forecast server. If the Forecast server is started without specifying -name, the logical name of the Forecast server is equal to the host name of the computer where the Forecast server is running.
-all	Directs a query or command to all Forecast servers. -all cannot be used with -M.

Parameter	Description
-cmdstr	Specifies a text string to be sent to the Forecast server. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" "). -cmdstr cannot be used with -reg or -cmd. The usage of -cmdstr is discussed in System configuration. The following commands are available: ▪ ORDER_ METHOD_CLEAN - clean all the info or just the info for one job key in the USER_DAILY_ZOS folder ▪ Syntax: ORDER_ METHOD_CLEAN <*CTM name*> [*dsn*] [*sched_folder*] [*memname*] ▪ ORDER_ METHOD_DATA - the order method definitions (job ordered, folder list or order method list) information gathered in the Control-M/EM database can be exported to a CSV file. The file can contain the following data: ▪ Data center name ▪ Lib name of job ▪ Folder name of job ▪ Job name ▪ Order type ▪ Target type ▪ Lib name of folder ▪ Folder name ▪ Task mask ▪ RBC mask ▪ ODAT ▪ Force ▪ Order method

Accessing the Configuration Management server from the ctl utility

This procedure describes how to access the Configuration Management server from the ctl utility, which enables you to send simple requests to the Configuration Management server.

➢ To access the Configuration Management server:

- Type the following command:

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
```

```
 -C CMS
{-M <computerName> |-name <logicalName> | -all}
{-reg |
 -cmd stop |
 -cmd life_check |
 -cmdstr "<commandString>"}
[-timeout <<responseTimeout> (seconds)>]
[-diagon]
```

For more information, see ctl-Configuration Management server parameters (on page 360).

ctl-Configuration Management server parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349) . The parameters specific to the Configuration Management Server are described in the following table:

Parameter	Description
-C CMS	Configuration Management Server to which the command is directed.
-M	Specifies a computer name. This name is used to identify the computer to which the Configuration Management Server belongs. -M cannot be used with -all. When -M is specified, the request is sent to the GUI Server whose name is equal to the value indicated with -M.
-name	Logical name of the Configuration Management Server. If the Configuration Management Server is started without specifying -name, the logical name of the Configuration Management Server is equal to the host name of the computer where the Configuration Management Server is running.
-all	Directs a query or command to all Configuration Management Servers. -all cannot be used with -M.
-cmdstr	Specifies a text string to be sent to the Configuration Management Server. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" "). -cmdstr cannot be used with -reg or -cmd. For more information about the -cmdstr parameter, see the System configuration.

Accessing the Configuration Agent from the ctl utility

This procedure describes how to access the Configuration Agent from the ctl utility, which enables you to send simple requests to the Configuration Agent server.

➢ To access the Configuration Agent:

- Type the following command

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
-C Config_Agent {-M <Computer_Name> | -all}
 {-reg |
  -cmd stop |
  -cmd life_check |
  -cmd shutdown} |
  -cmdstr "<commandString>"}]
 [-timeout <<responseTimeout> (seconds)>]
 [-diagon]
```

For more information, see ctl-Configuration Agent parameters (on page 361).

ctl-Configuration Agent parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the Configuration Agent are described in the following table:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-C Config_Agent</td><td colspan="2">Configuration Agent to which the command is directed. The Configuration Agent controls Control-M for Databases components on the host computer.</td></tr>
<tr><td>-M</td><td colspan="2">Specifies a computer name. This name is used to identify the computer to which the Configuration Agent belongs. -M cannot be used with -all.</td></tr>
<tr><td>-all</td><td colspan="2">Directs a query or command to all Configuration Agents.
-all cannot be used with -M.</td></tr>
<tr><td rowspan="2">-cmd</td><td colspan="2">Indicates a command to be performed by the Configuration Agent. -cmd cannot be used with -reg.</td></tr>
<tr><td>shutdown</td><td>Stops the Configuration Agent, and all components that the Configuration Agent administers, without changing their configurations. This command cannot be specified with other commands in the same run of the ctl utility.</td></tr>
</table>

Parameter	Description
-cmdstr	Specifies a text string to be sent to the Configuration Agent. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" "). -cmdstr cannot be used with -reg or -cmd. For more information about the -cmdstr parameter, System configuration.

Acessing the BMC Batch Impact Manager server from the ctl utility

This procedure describes how to access the BMC Batch Impact Manager server from teh ctl utility, which enables you to send simple requests to the BMC Batch Impact Manager server.

➢ To access the BMC Batch Impact Manager server:

- Type the following command:

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
  -C BIM
  {-M <computerName>
  -name <logicalName>}
 {-reg |
  -cmd stop |
  -cmd life_check |
  -cmdstr "<Command_String>"}
 [-timeout <<responseTimeout> (seconds)>]
 [-diagon]
```

For more information, see ctl-BMC Batch Impact Manager server parameters (on page 363).

ctl-BMC Batch Impact Manager server parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the BMC Batch Impact Manager server are described in the following table:

Parameter	Description
-C BIM	Indicates that the command is directed to a Batch Impact Manager server (if this component is installed at your site).
-M	Specifies a computer name. This name is used to identify the computer to which the Batch Impact Manager server belongs.
-name	Logical name of the Batch Impact Manager server. If the Batch Impact Manager server is started without specifying -name, the logical name of the Batch Impact Manager server is equal to the host name of the computer where the Batch Impact Manager server is running.
-cmdstr	Specifies a text string to be sent to the Batch Impact Manager server. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" "). -cmdstr cannot be used with -reg or -cmd. Its usage is discussed in System configuration.

Accessing the Control-M Self Service server from the ctl utility

This procedure describes how to access the Control-M Self Service server from the ctl utility, which enables you to send simple requests to the Control-M Self Service server.

➢ To access the Control-M Self Service server:

- Type the following command:

```
[em] ctl
[{-U <emUser> -P <emPass>} |-pf <passwordFile>]
  -C Self Service Server
  {-M <computerName>
  -name <logicalName>}
 {-reg |
  -cmd stop |
  -cmd life_check |
  -cmdstr "<Command_String>"}
 [-timeout <<responseTimeout> (seconds)>]
```

```
    [-diagon]
```

For more information, see ctl- Self Service server parameters (on page 364).

ctl- Self Service server parameters

The parameters common to all ctl commands are described in Parameters common to all ctl commands (on page 349). The parameters specific to the Self Service Server are described in the following table:

Parameter	Description
-C Self Service Server	Indicates that the command is directed to a Self Service Server (if this component is installed at your site).
-M	Specifies a computer name. This name is used to identify the computer to which the Self Service Server belongs.
-name	Logical name of the Self Service Server. If the Self Service Server is started without specifying -name, the logical name of the Self Service Server is equal to the host name of the computer where the Self Service Server is running.
-cmdstr	Specifies a text string to be sent to the Self Service Server. If the text string contains spaces or tabs, it must be enclosed with double quotation marks (" "). -cmdstr cannot be used with -reg or -cmd. Its usage is discussed in System configuration.

ctl example

This example describes specifying a specific server component and computer. The following table lists examples showing how you can ensure that only specific server components and computers are selected. In the examples, only the relevant section of the code is displayed.

To do this ...	Specify this ...
Direct the ctl command to the Gateway for data center ctm_main.	em ctl -U user01 -P pass01 -C Gateway -dc ctm_main
Direct the ctl command to all Global Conditions Servers.	em ctl -U user01 -P pass01 -C GCS -all
Direct the ctl command to the Global Conditions Server of the computer named wip78.	em ctl -U user01 -P pass01 -C GCS -M wip78
Direct the ctl command to the GUI Server on the computer with the logical name of gsr01.	em ctl -U user01 -P pass01 -C GUI_Server -name gsr01

orbadmin

The orbadmin (CORBA administration) utility can be used to manage the Naming Service process and the CORBA configuration file. To run the orbadmin utility, see Running the orbadmin utility (on page 365).

Control-M/EM uses an XML CORBA configuration file that defines CORBA parameters for CORBA components. The parameters and values in this file can be initialized and modified by the orbconfigure GUI (see the description of the orbconfigure CORBA configuration GUI in C*ontrol-M Workload Automation Administration*), and by the orbadmin utility (described here).

Running the orbadmin utility

This procedure describes how to run the obadmin utility, which enables you to manage the Naming Service process and the CORBA configuration file.

➢ To run the orbadmin utility:

- Type one of the following commands to invoke the orbadmin utility:
  - `orbadmin domain <domainCommand>`
  - `orbadmin ns <namingServiceCommand> [<option>]`
  - `orbadmin scope <scopeCommand> [<scopeName>]`
  - `orbadmin variable <variableCommand> [-scope <scopeName>] -value <value> | <varName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the orbadmin parameters, see orbadmin parameters (on page 367).

orbadmin parameters

The following table lists the orbadmin parameters:

Command format	**Usage**
domain show	Displays all parameters and values in the configuration file
ns list [-ior]	Lists all registered objects in the Naming Service with endpoint information. If -ior is specified, IOR references of the registered objects are printed to the console.
ns probe	Checks if the local Naming Service is configured and registered.
ns register	(*Windows only*) Specifies Naming Service values in the registry and installs the Naming Service as a Windows service.
ns remove	(*Windows only*) Removes Naming Service values from the registry and uninstalls the Windows service.
ns resolve	Resolves the Naming Service IOR and prints it to the console.
ns start	Starts the local Naming Service.
ns status	Checks the status of the configured Naming Service process. Possible status values: Running, Stopped, and Not installed.
ns stop [-local]	Stops the Naming Service. **-local** is required if a remote Naming Service is running and you want to stop the local Naming Service.
scope create <*scopeName*>	Creates a new scope in the configuration file
scope list	Displays all the scopes in a configuration file
scope remove <*scopeName*>	Removes a scope and its parameters from the file
scope show <*scopeName*>	Displays parameters and their values in the scope
variable create -scope *scopeName* [-value <*value*>]	Creates a parameter (with or without a value) in the specified scope. If that scope does not exist, it is created. If the value contains a blank, enclose the value in double quotes.
variable modify [-scope <*scopeName*>] -value <*value*>	Modifies a parameter value (in the specified scope) of the configuration domain. If the parameter does not exist, it is added automatically. If the scope is not specified, the parameter is modified in every scope that contains it. If the specified value contains a blank, enclose the value in double quotes.

Command format	Usage
variable remove [-scope *scopeName*] <*varName*>	Removes specified parameter from the XML configuration file. If the scope is not specified, the parameter is removed from all the scopes that contain it.
variable show -scope *scopeName* <*varName*>	Displays the specified parameter and its value in the specified (or default) scope. If the parameter does not exist in the specified scope, the default scope is searched.

orbadmin examples

The following are examples of outputs and examples for some of the orbadmin utility commands:

ns status sample output:

```
Naming service status: Running on remote machine, vered:13075
```

ns resolve sample output:

```
IOR:0100000001000000000000000001000000000000002400000000010102cd060000
0076657265640013330b0000004e616d6553657276696365cd00000000
```

ns list sample output:

```
BMC Software: naming context

 EM: naming context

  vered: naming context

    GSR: naming context

     XMLInvoker: object reference:Protocol: IIOP, Endpoint:
172.16.131.242:59020

      EMSystemParameters: object reference: Protocol: IIOP,
Endpoint: 172.16.131.242:59020

     CollectionRepository: object reference: Protocol: IIOP,
Endpoint: 172.16.131.242:59020

     SchedEntityFieldsDesc: object reference: Protocol: IIOP,
Endpoint: 172.16.131.242:59020

     DefDB: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020

     ViewRepository: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020

     CTMRepository: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020
```

```
     BimProxy: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020

     FilterRepository: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020

     HierarchyRepository: object reference: Protocol: IIOP,
Endpoint: 172.16.131.242:59020

     PlayerControl: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020

     CommAdminDB: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020

     UsersManager: object reference: Protocol: IIOP, Endpoint:
172.16.131.242:59020
```

ns list -ior sample output:

```
BMC Software: naming context

EM: naming context

vered: naming context

GSR: naming context

 XMLInvoker:<IOR:010000001a00000049444c3a4543534150492f584d4c496e7
66f6b65723a312e30000000100000000000000074000000010102cd0f000000313
7322e31362e3133312e32343200658ce672762300000014010f004e535442301d6
9000952f90000000100000001000000a0000000100000000bcd020000000000000
00800000000000000054414f00010000001400000000000000000100010000000000
001010900000000>
```

scope create

The following command creates a scope named **test:**

```
orbadmin scope create test
```

After creating a new scope, use the `orbadmin variable create` command to add parameters to the new scope.

scope list

To display all the scopes in the XML configuration file:

orbadmin scope list

default

GSR

GUI

XmlUtils

ns

scope show

This command displays the contents of the default configuration scope:

orbadmin scope show default

```
-ORBInitRef = NameService=corbaloc::1.2@host:12345/Name Service
-ThreadPoolSize = 10
```

variable create

This example creates a variable named **-PreferIPMask** in the scope Default.

orbadmin variable create -scope default -value 172.16.0.0 -PreferIPMask

variable modify

This example modifies the filename for the **pid** of the naming service.

orbadmin variable modify -scope ns -value ns2.pid -p

variable show

This example displays a parameter in the default configuration scope:

orbadmin variable show orb -ThreadPoolSize

```
-ThreadPoolSize = 10
```

This example shows the same parameter as it is set for the GUI Server in the configuration scope GSR:

orbadmin variable show -scope GSR -ThreadPoolSize

```
-ThreadPoolSize = 15
```

This example shows the same parameter as it is set for the CLI, although the configuration scope CLI does not contain it:

orbadmin variable show -scope CLI -ThreadPoolSize

```
-ThreadPoolSize = 10
```

Control-M/Server utilities

This table lists the Control-M/Server utilities for communication, startup, and troubleshooting.

Utility Type	Description
ctm_agstat (on page 372)	The ctm_agstat utility enables you to list or update the status of an agent.
ctm_diag_comm (on page 374)	The ctm_diag_comm utility generates a report about the connection details between the specified Control-M/Agent or remote host and Control-M/Server.
ctmgetcm (on page 376)	The ctmgetcm utility is used to collect, store and display application server information from Control-M/Agents (version 6.1.01 or later).
ctmhostmap (on page 378)	The ctmhostmap utility manages the mapping of remote host computers to agents and the conversion of Control-M/Agents to remote host computers.
ctmhostgrp (on page 384)	The ctmhostgrp utility is used to maintain and view host groups. This utility provides the command line facility for running the options available from the Host Group Menu.
ctmping (on page 386)	The ctmping utility tests, configures, and reports on the connection and availability between Control-M/Server and Control-M/Agents or remote host computers.
ctmshout (on page 390)	The ctmshout utility sends a message to the specified user or destination using the specified severity level.
ctmshtb (on page 393)	The ctmshtb utility specifies the active Shout Destination folder.
ctmspdiag (on page 393)	The ctmspdiag utility is a tool to print or erase diagnostic messages recorded from stored procedures (SPs) in the Control-M/Server database.
ctmsuspend (on page 396)	The ctmsuspend utility suspends and restores Control-M for Databases non-communication processes for mass uploads and downloads from Control-M/EM. During suspension mode, Control-M inactivates its job processing functions by suspending the TR, SL, NS, LG, and WD processes.
init_prflag (on page 397)	The init_prflag utility resets sleep times and trace levels for Control-M/Server processes.

Utility Type	Description
Running the shut_ca utility (on page 399)	The shut_ca utility is used to shut down the Control-M/Server Configuration Agent.
Running the shut_ctm utility (on page 399)	The shut_ctm utility is used to shut down Control-M/Server and its processes.
Running the show_ca utility (on page 399)	The show_ca utility is used to display the status of the Control-M/Server Configuration Agent.
Running the shutdb utility (on page 400)	The shutdb utility is used to shut down the SQL database server.
Running the start_ca utility (on page 400)	The start_ca utility is used to start up the Control-M/Server Configuration Agent.
Running the start_ctm utility (on page 400)	The start_ctm utility is used to start Control-M/Server.
Running the startdb utility (on page 401)	The startdb utility is used to start the SQL database server.

ctm_agstat

The ctm_agstat utility enables you to list or update the status of an agent. For more information, see communication status of agents and remote hosts in Component management. To run the ctm_agstat utility, see Running the ctm_agstat utility (on page 372).

Running the ctm_agstat utility

This procedure describes how to run the ctm_agstat utility, which enables you to list or update the status of an agent.

➢ To run the ctm_agstat utility:

- Type the following command:

```
ctm_agstat
{ -LIST <agentName> | -UPDATE <agentName> {AVAILABLE | DISABLED} }
 [ -DEBUG <debug level 1-5> ]
 [ -QUIET ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctm_agstat parameters, see ctm_agstat parameters (on page 373).

ctm_agstat parameters

The following table describes the parameters of the ctm_agstat utility:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-LIST</td><td colspan="2">Lists the current status of the specified agent.
This parameter is mandatory if the -UPDATE parameter is not specified.</td></tr>
<tr><td rowspan="4">-UPDATE</td><td colspan="2">Changes the status of the specified agent to the specified status.
This parameter is mandatory if the -LIST parameter is not specified
The permissions of the AGSTAT directory must be modified to allow access to users who will run the ctm_agstat utility with the -UPDATE parameter.</td></tr>
<tr><td>*<agentName>*</td><td>Host name of the agent to be listed or updated. This name must be specified for -LIST and -UPDATE parameters.</td></tr>
<tr><td>AVAILABLE</td><td>Status where Control-M/Server is able to communicate with the specified Control-M/Agent.</td></tr>
<tr><td>DISABLED</td><td>Status where Control-M/Server ignores (does not attempt to communicate with) the specified Control-M/Agent.</td></tr>
<tr><td>-DEBUG <debug level></td><td colspan="2">Set the required diagnostic level. Valid values: 0 (no diagnostics) to 5 (highest level of diagnostics).</td></tr>
<tr><td>-QUIET</td><td colspan="2">If this parameter is specified, the utility runs without displaying output messages.</td></tr>
</table>

ctm_agstat examples

The following are examples of the ctm_agstat utility commands:

To display the current status of agent CMAGENT, type:

```
ctm_agstat -LIST CMAGENT
```

To change the status of agent CMAGENT to DISABLED, type:

```
ctm_agstat -UPDATE CMAGENT DISABLED
```

ctm_diag_comm

The ctm_diag_comm utility generates a report about the connection details between the specified Control-M/Agent or remote host and Control-M/Server. You can either specify the host name of a computer where Control-M/Agent is installed, or the host name of a computer with which you want Control-M/Server to communicate. To run the ctm_diag_comm utility, see Running the ctm_diag_comm utility (on page 374).

Running the ctm_diag_comm utility

This procedure describes how to run the ctm_diag_comm utility, which enables you to

➢ To run the ctm_diag_comm utility:

- Do one of the following:
  - Type the following command from the command prompt, to run the utility interactively:

    ```
    ctm_diag_comm
    ```

  You are prompted for the host name of the Control-M/Agent or remote host.

  - Type the following command from the command prompt, to run the utility as a single command:

    ```
    ctm_diag_comm <agent> | <remoteHost>
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctm_diag_comm parameters, see ctm_diag_comm parameters (on page 374).

ctm_diag_comm parameters

The following table lists the ctm_diag_comm utility parameters:

Parameter	Description
<agent>	Host name of the computer where Control-M/Agent is installed
<remoteHost>	Name of the specified remote host with which Control-M/Server should attempt to communicate.

ctm_diag_comm example

The following is an example of ctm_diag_comm report:

Assume that Control-M/Server is installed on the UNIX computer with host name taurus. Control-M/Agent has not yet been defined in the Control-M/Server database, so when ctm_diag_comm is invoked, it will try to generate a report connecting it as an agent or remote host. The report below is displayed after invoking the following command:

```
Control-M/Server to Control-M/Agent Communication Diagnostic Report
-------------------------------------------------------------------
CTMS User Name                  : ctm630sd
 CTMS Directory                  : /home1/ctm630sd/ctm
CTMS Platform Architecture      : Solaris
 CTMS Installed Version          : DRCTV.6.3.01
CTMS Local IP Host Interface Name : taurus
Server-to-Agent Port Number     : 12666
Agent-to-Server Port Number     : 7420
Server-Agent Comm. Protocol     : TCP
 Server-Agent Protocol Version   : 07
Server-Agent Connection mode    : Transient
Agent Platform Name             : mars
Agent Status                    : Unknown
Agent known Type                : Undefined
UNIX ping to Agent or Remote host : Succeeded
CTMS ping to Agent or Remote host : Succeeded
CTMS Ping mars as Regular Agent
==============================
Agent [mars] responded with "Agent not available."
CTMS Ping mars as Remote Host
============================
Remote host [mars] is available through Agent [nova]
Connection protocol : SSH
Port number         : 22
Encryption method   : BLOWFISH
Compression         : No
```

ctmgetcm

The ctmgetcm utility is used to collect, store and display application server information from Control-M/Agents. To run the ctmgercm utility, see Running the ctmgetcm utility (on page 376).

- When the action parameter is set to GET, application server information is collected, stored in a folder in the Control-M/Server database, and displayed.
- When the action parameter is set to VIEW, previously stored application server information is displayed.

Control-M information is updated only after ctmgetcm is run, or each time ctmgetcm is reconfigured.

Running the ctmgetcm utility

This procedure describes how to run the ctmgetcm utility, which enables you to collect, store and display application server information from Control-M/Agents.

➢ To run the ctmgetcm utility:

Do one of the following:

- Type the following command from the command prompt, to run the utility interactively:

  ```
  ctmgetcm
  ```

  You are prompted for the required parameters as if you had selected the View Host ID Details option of the Control-M Main Menu.

- Type the following command from the command prompt, to run the utility as a single prompt:

  ```
  ctmgetcm -HOST agent -APPLTYPE OS -ACTION <get|view>
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmgetcm parameters, see ctmgetcm parameters (on page 377).

ctmgetcm parameters

The following table lists the ctmgetcm utility parameters:

Parameters	Description	
-HOST	Host name of the agent computer.	
-APPLTYPE	Name of the application server (for example, SAP). A wildcard character can be used to specify more than one application: Specify * to retrieve information for all applications. Specify O* to retrieve information for all applications beginning with O (for example, ORA or OS). OS can be specified to return information about the Application Add-on for the current operating system.	
-ACTION	Indicates the action that the ctmgetcm utility should perform. Valid values are:	
	GET	Collect and display updated application information from the specified Control-M/Agent. The information collected is stored in the Control-M/Server database.
	VIEW	Display application server information that was previously collected from the specified agent computer. This action will display only information that was retrieved previously using a GET action.

ctmgetcm example

The following are examples of the ctmgetcm utility commands:

Specify the following command to view existing information for all applications on the Control-M/Agent sahara computer:

```
ctmgetcm -host sahara -appltype "*" -action VIEW
```

Output similar to the following is displayed:

```
HOST            APPLTYPE        APPLVER     CMVER
-------        ---------       --------    ------
sahara            ORA          7           2.0.00
sahara            SAP          4.5         2.0.01
sahara            OS           SOLARIS     1.0.00
```

Specify the following command to view existing information for all applications with prefix O on the Control-M/Agent sahara computer:

```
ctmgetcm -host sahara -appltype O* -action VIEW
```

Output similar to the following is displayed:

```
HOST           APPLTYPE       APPLVER    CMVER
-------      ---------      --------   ------
sahara          ORA          7          2.0.00
sahara          OS           SOLARIS    1.0.00
```

Specify the following command to update the Control-M/Server database with new information for all applications with prefix O on the Control-M/Agent sahara computer:

```
ctmgetcm -host sahara -appltype O* -action GET
```

The Control-M/Server database is updated and output similar to the following is displayed:

```
HOST           APPLTYPE       APPLVER    CMVER
-------      ---------      --------   ------
sahara          ORA          7          2.0.00
sahara          ORA          8          2.0.00
sahara          OS           SOLARIS    1.0.00
```

ctmhostmap

The ctmhostmap utility manages the mapping of remote host computers to agents and the conversion of Control-M/Agents to remote host computers. This utility can be run from the command line, or by using the Control-M Configuration Manager. For more information about using the Control-M Configuration Manager, see Introduction to Control-M Configuration Manager. To run the ctmhostmap utility, see Running the ctmhostmap utility (on page 378).

Each computer that is defined as a remote host is listed in the Control-M/Server database. The ctmhostmap utility enables you to manage the entries in the database.

Running the ctmhostmap utility

This procedure describes how to run the ctmhostmap utility, ehich enables you to manage the mapping of remote host computers to agents and the conversion of Control-M/Agents to remote host computers.

➢ To run the ctmhostmap utility:

- Type one of the following commands to run the ctmhostmap utility:

  -
    ```
    ctmhostmap -action add [-force] -host <remoteHost>
    -agent <agentsList>
    -protocol SSH|WMI [-sshport <SSHportNumber>
    -sshalg BLOWFISH|AES|DES|3DES
    -compression <SSH compression Y/N>] [-outputdir <WMIoutputDirectory>]
    ```

-
  ```
  ctmhostmap -action update -host <remoteHost> [-agent <agentsList>]
  [-protocol SSH|WMI]
  [[-sshport <SSHportNumber>
  -sshalg BLOWFISH|AES|DES|3DES
  -compression <SSH compression Y/N>] [-outputdir <WMIoutputDirectory>]]
  ```
- `ctmhostmap -action delete -host <remoteHost>`
- `ctmhostmap -action list [-host <remoteHost>]`
- `ctmhostmap help`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmhostmap parameters, see ctmhostmap parameters (on page 381).

ctmhostmap Actions

The following table describes the actions in the ctmhostmap utility:

Action	Description
add	Specifies the details of the remote host computer that is being defined in the Control-M/Server database. Control-M/Server discovers the specified computer using default remote host map settings if the following is specified without further parameters: `ctmhostmap -action add -host <remoteHost>` Separate the agent names with a semicolon (;) when adding more than one agent. For UNIX: If more than one Control-M/Agent is being added, the entire list must be enclosed in quotation marks, for example "jupiter;andromeda;taurus".
update	Modifies the details of the specified remote host computer in the Control-M/Server database. Separate the agent names with a semicolon when adding more than one agent. For UNIX: If more than one Control-M/Agents are being added, the entire list must be enclosed in quotation marks, for example "jupiter;andromeda;taurus". Control-M/Server replaces the existing settings of a remote host with the settings of Default if the following command is specified: `ctmhostmap -action update  -host "<Default>"`
delete	Deletes the details of the specified remote host computer in the Control-M/Server database.
list	Lists the details of the specified remote host computer in the Control-M/Server database. The resultant list includes all remote host computers and their statuses, except when **list** is specified with **-host** (see description below).
help	Displays the usage of the ctmhostmap utility.

ctmhostmap parameters

The following table lists the ctmhostmap utility parameters:

Parameter	Description
-force	Override a regular agent computer that has the same name as the specified computer. Optional. This option is used in order to convert regular agent to remote host. For more information, see the appendix in the in Defining a remote host.
-host	Specifies the name of the remote host computer or <Default>. The name of the host cannot exceed 50 characters. This parameter is: ▪ mandatory when specified with add, update, and delete ▪ optional when specified with list <Default > is not applicable if the delete action is specified. If -host is specified with -list, the remote host is not <Default>, and the status is not Discovering, then the details of the specified remote host are displayed. The output includes the following: ▪ all the remote host definition parameter values ▪ remote host status ▪ status of each agent, showing which remote host is defined to be available through it To view the "Default Remote Settings", specify the following command: `ctmhostmap -action list  -host "<Default>"`
-agent	List of agent names, separated by semi-colons (;). For UNIX: Enclose the entire list in quotation marks. This parameter is: ▪ mandatory when specified with add ▪ optional when specified with update For example, to list more than one Control-M/Agent, the entire list must be separated by semi-colons, for example pluto;mars;saturn. For UNIX, the list would be: "pluto;mars;saturn".

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="6">-protocol</td><td colspan="2">Indicates which protocol is used by the agent to execute jobs on the remote host computer. Mandatory. Valid values are:
▪ SSH – Secure Shell (SSH) for UNIX or Windows computers
▪ WMI – Windows Management Instrumentation (WMI) for Windows computers
If SSH is specified, then the following parameters must be specified:</td></tr>
<tr><td>-sshport</td><td>Specifies the SSH port number that the SSH daemon is listening to on the remote host computer. Valid value: 22 or an integer from 1024 through 65535</td></tr>
<tr><td>-sshalg</td><td>Indicates which SSH encryption algorithm is being used. Valid values are:
▪ BLOWFISH
▪ AES
▪ DES
▪ 3DES</td></tr>
<tr><td>-compression</td><td>Valid values are:
▪ Y – compression is used
▪ N – compression is not used</td></tr>
<tr><td colspan="2">If WMI is specified, then the outputdir parameter must be specified.
The outputdir must either be prefaced with double back-slashes (for example <b>d:\\output_dir</b>), or be enclosed with quotation marks (for example <b>"d:\output_dir"</b>).</td></tr>
<tr><td>-outputdir</td><td>Indicates the directory used for the OUTPUT files created by jobs that have been submitted. The name of the outputdir cannot exceed 1,024 characters.
Configure the specified directory on the remote host as a shared directory, with the shared name of <b>OUTPUT</b>.</td></tr>
</table>

ctmhostmap examples

The following are examples of the ctmhostmap utility commands:

To add remote host mars to the Control-M/Server database, mapped through Control-M/Agents pluto, neptune, and venus, using SSH protocol, run the following command:

ctmhostmap -action add -host mars -agent "pluto;neptune;venus" -protocol ssh -sshport 54 -sshalg des -compression N

The following message is displayed:

```
Remote host added successfully.
```

To add remote host saturn, mapped through Control-M/Agent jupiter, using WMI protocol, run the following command:

```
ctmhostmap -action add -host saturn -agent jupiter -protocol wmi
-outputdir d:\\ctm\\data
```

The following message is displayed:

```
Remote host added successfully.
```

As described above, but the OUTPUT directory contains spaces. Run the following command:

```
ctmhostmap -action add -host saturn -agent jupiter -protocol wmi
-outputdir "c:\ctm eur\data"
```

The following message is displayed:

```
Remote host added successfully.
```

To add remote host saturn, mapped through Control-M/Agents jupiter, andromeda, and taurus using SSH protocol, run the following command:

```
ctmhostmap -action add -host mars -agent "jupiter;andromeda;taurus"
-protocol ssh -sshport 22 -sshalg blowfish -compression N
```

The following message is displayed:

```
Remote host added successfully.
```

When modifying an existing entry, only the parameters that are being updated are mandatory. To change the SSH port number of remote host mars from 54 to 48, and to change the SSH algorithm from DES to 3DES, run the following command:

```
ctmhostmap -action update -host mars -sshport 48 -sshalg 3des
```

The following message is displayed:

```
Remote host updated successfully.
```

To delete remote host mars from the Control-M/Server database, specify the following command:

```
ctmhostmap -action delete -host mars
```

The following message is displayed:

```
Action ‘delete’ ended successfully.
```

To display a list of remote hosts, run the following command:

```
ctmhostmap -action list
```

The following report is displayed:

```
Remote Host          Status
orion                Available
taurus               Unavailable
pegasus              Unavailable
Action ‘list’ ended successfully.
```

To display the details of remote host orion, run the following command:

```
ctmhostmap -action list -host orion
```

The following report is displayed:

```
Remote host ‘orion’ settings:
Protocol        : SSH
Port Number     : 22
Encryption      : BLOWFISH
Compression     : NO
Agents          : (comet) (meteor) (cyborg)
Remote Host Status      : Available
Agents Statuses         : (comet: Available) (meteor: Available)
(cyborg: Available)
Action ‘list’ ended successfully.
```

ctmhostgrp

The ctmhostgrp utility is used to maintain and view host groups. This utility provides the command line facility for running the options available from the Host Group Menu. To run the ctmhostgrp utility, see Running the ctmhostgrp utility (on page 384).

The Host Group menu is used to maintain and view host groups. Host groups are used by the Control-M/Server load-balancing function. For additional instructions, see Host group management.

Running the ctmhostgrp utility

This procedure describes how to run the ctmhostgrp utility, which enables you to maintain and view host groups.

➢ To run the ctmhostgrp utility:

- Type the following command to run the ctmhostgrp utility:

```
ctmhostgrp
[ -LIST                                |
     -EDIT   -HOSTGRP <hostGroup>
                 -APPLTYPE <applicationType>
```

```
                  [ -VIEW              |
                    -ADD      <hostid> |
                    -DELETE   <hostid>   ]     |
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmhostgrp parameters, see ctmhosgrp parameters (on page 385).

ctmhosgrp parameters

The following table lists the ctmhostgrp utility parameters:

Variable	Description		
-LIST	Displays a list of all existing host groups.		
-EDIT	Views, creates, or modifies a host group.		
	-HOSTGRP	Specifies which host group to be viewed, created, or modified.	
	-APPLTYPE	Specify the application with which the host group is associated. Valid values are: ■ OS – None Application Add-on (Default) ■ SAP – SAP Applications ■ OAP – Oracle Applications ■ Any other application type defined by the user. For more information, see the *Software Development Guide for Application Add-ons*.	
		-VIEW	Displays the host IDs in the specified host group.
		-ADD	Prompts for the name of a host ID to add to the specified group. The selected host ID must be able to run jobs associated with the application type specified for this host group.
		-DELETE	Prompts for the name of a host ID to delete from the specified group.
-DELETE	Prompts for the name of the host group to delete from the specified group.		

ctmping

The ctmping utility tests, configures, and reports on the connection and availability between Control-M/Server and Control-M/Agents or remote host computers. To run the ctmping utility, see Running the ctmping utility (on page 386).

ctmping can be included in the Watchdog process. For more information, see Watchdog process parameters.

This utility can check if an agent or remote host is down and, if required, register it in the database as unavailable. When the agent or remote host again becomes available, the state is changed and information about it is gathered by a Control-M/Server process.

You cannot ping the agent if the agent is upgrading. When the agent becomes available again, then you can ping the agent. The following error message appears:

```
Cannot ping the agent. Agent is upgrading.
```

Running the ctmping utility

This procedure describes how to run the ctmping utility, which enables you to test, configure, and report on the connection and availability between Control-M/Server and Control-M/Agents or remote host computers.

➢ To run the ctmping utility:

- Type the following command:

```
ctmping -HOSTID <hostName> [ -HOSTTYPE <REGULAR|REMOTE> ]
 [ -HOSTID   <hostName> [ -HOSTTYPE <REGULAR|REMOTE> ] ]
 [ -FILE <inputFile> ]
 [ -DISCOVER <y|n>]
 [ -DEBUG    <debugLevel> ]
 [ -QUIET ]
[ -FULLDETAILS ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmping parameters, see ctmping parameters (on page 387).

ctmping parameters

The following table lists the ctmping utility parameters:

Parameter	Description
-HOSTID	Host name of the agent or remote host computer to be pinged (tested). At least one Host ID must be specified for each run of the ctmping utility. Additional Host IDs can optionally be specified to enable a single run of this utility to test communication with more than one agent computer.
-HOSTTYPE	Describes the type of computer that is being pinged. REGULAR – A computer that is registered as an agent in the Control-M/Server database. Control-M/Agent is installed on this computer. REMOTE – A computer that is registered as a remote host in the Control-M/Server database. It is not necessary for a Control-M/Agent to be installed on this computer. A computer that has a Control-M/Agent installed on it, can also be registered as a remote host. If you do not specify a HOSTTYPE: ▪ if the host ID of the computer is already defined, the HOSTTYPE that was defined is used again ▪ if the host ID of the computer was not used previously, then REGULAR is used as the default ▪ in the event that the type is unknown, that is, still in host discovery state, the utility is unable to complete the request. Wait for the discovery process to complete, then run the utility again. If you specify a HOSTTYPE: ▪ if the host ID is already defined in Control-M/Server and it conflicts with the earlier definition (for example, it was defined earlier as REGULAR and you are now specifying REMOTE), then the utility will not be able to complete the request. If the intention was to convert the defined host type, then for more information about how to convert a computer from one type to another, see ctmhostmap (on page 378) and Defining a remote host. ▪ if the host ID is already defined in Control-M/Server and the HOSTTYPE being specified now is same as that of the earlier definition, then the utility will use the specified HOSTTYPE for the host ID. ▪ if the host ID has not been previously defined in Control-M/Server, then the utility will use the specified HOSTTYPE for the host ID.

Parameter	Description

-FILE	Full path and name of a file containing a list of host computers to be pinged.
	Each line in the specified file contains:
	▪ the name (host ID) of the agent or remote host (mandatory)
	▪ the HOSTTYPE (optional)
	The delimiter between the name and the HOSTTYPE is a blank or any number of blank spaces.
	Where the computer type is not specified, a discovery process attempts to determine the type to be used. See -HOSTTYPE above.
	Assume the following computers must be pinged:
	`Name Type`
	`dime not specified`
	`comet Control-M/Agent`
	`mars remote host computer`
	The text below specifies the above details in a file:
	`dime (HOSTTYPE is not specified)`
	`comet REGULAR`
	`mars REMOTE`

-DISCOVER	Indicates whether to update the database. ▪ Y – Update the database with information gathered by the utility ▪ N – Do not update the database If -DISCOVER is not specified, ctmping performs an automatic discovery, as follows: If an agent is ▪ available – the Control-M/Server database is updated with the details of the agent or remote host ▪ unavailable, for a new agent or remote host in the Control-M/Server database, a message is displayed, asking if you want to add the discovered agent to the Control-M/Server database Default: automatic discovery

Parameter	Description
-DEBUG	Activates a debug trace at the specified level. Valid levels: 0 through 5. Default: 0. Performance is slower when operating in debug mode. BMC recommends that you activate debug mode only when requested by Technical Support and use the specified level.
-QUIET	Suppresses display of utility output.
-FULLDETAILS	Generates a status report of the communication parameters of each specified agent or remote host. Where an agent is unavailable, the report displays the reason why the agent is unavailable. Where a remote host is available it displays the connection information.

ctmping examples

The following are examples of the ctmping utility commands:

To connect and perform a communication test with the agent jacklin, specify the following command:

```
ctmping -hostid jacklin
```

The response is:

```
Agent: jacklin is alive
```

To attempt to connect and test communication with agent diana (that is currently down), specify the following command:

```
ctmping -hostid diana
```

The response is:

```
Agent: diana, Msg: Agent not available.
Add it to the database? y/n:
```

To connect and test communication with the agent jacklin and collect configuration information needed for the discovery process, specify the following command:

```
ctmping -hostid jacklin -discover y
```

The response is:

```
Agent: jacklin is alive
```

To connect and test communication with the agent jacklin and generate a debug trace without displaying the results on screen, specify the following command:

```
ctmping -hostid jacklin -debug 1 -quiet
```

Only the return code of the utility indicates if it was successful. The debug trace information is saved to the following file:

```
~<controlm_owner>/ctm_server/proclog/ping<PID>_<PID>.log
```

PID is the Process Identity number.

The agents, comet and mars, have been configured to connect to remote host dime. To generate a report showing the result of the test and the connection details, specify the following command:

```
ctmping -hostid dime -hosttype remote -fulldetails
```

The following response is displayed:

```
Remote host [dime] is available through Agent [comet]
Connection protocol : SSH
 Port number         : 22
 Encryption method   : Blowfish
 Compression         : No
Remote host [dime] is available through Agent [mars]
 Connection protocol : SSH
 Port number         : 22
  Encryption method   : Blowfish
 Compression         : No
```

Assume the **hostid_list.txt** text file contains the full path to the following host IDs:

- local agent jacklin
- remote host dime
- regular agent diana

To generate a report showing the result of pinging the host IDs specified in **hostid_list.txt**, specify the following command:

```
ctmping -file ~<controlm_owner>/ctm_server/hostid_list.txt
```

The response is:

```
Agent : jacklin is alive
Remote host : dime is alive
Agent: diana Msg: Agent not available
```

ctmshout

The ctmshout utility sends a message to the specified user or destination using the specified severity level. For information about Shout message destinations, see Shout destination management. For information about attaching the OUTPUT of a job to Shout messages, see the E-mail configuration parameters. To run the ctmshout utility, see Running the ctmshout utility (on page 391).

Each parameter name can be shortened to the minimum number of letters required to uniquely identify the parameter. For example: -ORDERID can be shortened to -O.

Shout messages can be sent to multiple destinations or users in the same command

Running the ctmshout utility

This procedure describes how to run the ctmshout utility, which enables you to send a message to the specified user or destination using the specified severity level.

➢ To run the ctmshout utility:

- Type one of the following command to invoke the ctmshout utility:

  -
    ```
    ctmshout
    -USER <userName> or -DEST <destinationName>
    -MESSAGE "<messageText>"
    [ -ORDERID <orderID>]
    [ -HOSTID <hostID>]
    [ -SEVERITY <severityLevel>]
    ```
  - `ctmshout -input_file <fullPathFileName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmshout parameters, see ctmshout parameters (on page 391).

ctmshout parameters

The following table lists the ctmshout utility parameters:

Parameter	Description
<username>	User ID that should receive the message. DEST and USER can be specified in the same ctmshout command.
<destinationName>	Logical destination device name or a valid destination name in the Shout Destination folder. DEST and USER can be specified in the same ctmshout command.
<message>	Free text to be sent to the destination (1 - 255 characters). If the text is more than one word, it must be enclosed in quotation marks.
<orderID>	Order ID of a job, as displayed in the Job Details window in Control-M/EM. Associates the message with a specific job in the Active Network.
<hostID>	Host ID of the agent computer. Used for messages whose destination is either a user in the data center or a user defined in the Shout Destination folder. If -ORDERID is also specified, this Host_ID overrides the Host ID specified in the job with that Order ID.
<severityLevel>	One letter character indicating the urgency of the message. Valid

Parameter	Description
	values: - R – Regular (Default) - U – Urgent - V – Very urgent
<fullPathFileName>	Name and full path of a file containing the utility parameters. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. The -input_file parameter enables you to: prepare and save files of utility parameters that can be reused. specify utility input longer than the number of characters allowed in the command line. `-input_file~<controlm_owner>/ctm_server/data/ctmshout_parms.txt`

ctmshout examples

The following are examples of the ctmshout utility commands:

The following command sends the message "File not found" to the Alerts window in Control-M/EM and associates it with a job whose Order ID is 1234:

```
ctmshout -ORDERID 1234 -USER EM \

-MESSAGE "File not found" -SEVERITY V
```

The same result is achieved by using the -input_file parameter as follows:

```
ctmshout -input_file
~<controlm_owner>/ctm_server/data/ctmshout_payroll.txt
```

The referenced file contains the following lines:

```
-ORDERID 1234

-USER EM

-MESSAGE "File not found"

-SEVERITY V
```

The following command sends the message "The weekly paycheck job has abended" to user John on agent computer diana:

```
ctmshout -HOSTID diana -USER John -MESSAGE
"The weekly paycheck job\ has abended" -SEVERITY V
```

The following illustrates the use of the ctmshout utility in a job script command to send the Shout message "Job started" to the Alerts window in Control-M/EM.

The job processing definition for a certain job contains the following Variable Assignment parameter:

```
%%PARM1 = %%ORDERID
```

The script used to execute the job contains the following command:

```
ctmshout -O $1 -USER EM -MESSAGE "Job started" \
-SEVERITY R
```

ctmshtb

The ctmshtb utility specifies the active Shout Destination folder. To run the ctmshtb utility, see Running the ctmshtb utility (on page 393).

You can add, delete, and modify Shout Destination folders using the ctmsys utility, described in ctmsys (on page 445) . The ctmsys utility can also be used to specify the active Shout Destination folder interactively.

The Shout Destination folder associates physical output destinations with logical destination names. The names are specified in Shout and Do Shout statements in job processing definitions and in the ctmshout utility. For more information, see Shout destination management.

By defining Control-M jobs that execute this utility at specified times, the active Shout Destination folder designation can be changed automatically according to the schedule that suits your installation.

Running the ctmshtb utility

This procedure describes how to run the ctmshtb utility, which enables you to specify the active Shout Destination folder.

➢ To run the ctmshtb utility:

- Type the following command to invoke the ctmshtb utility:

  ```
  ctmshtb <folder>
  ```

  <folder>:New Shout Destination folder name.

The following command sets the current active Shout Destination folder designation to SHIFTMAN:

```
ctmshtb SHIFTMAN
```

ctmspdiag

The ctmspdiag utility is a tool to print or erase diagnostic messages recorded from stored procedures (SPs) in the Control-M/Server database. ctmspdiag can also set or show the diagnostic request status of SPs. To run the ctmspdiag utility, see Running the ctmspdiag utility (on page 394).

Running the ctmspdiag utility

This procedure describes how to run the ctmspdiag utility, which enables you to print or erase diagnostic messages recorded from stored procedures (SPs) in the Control-M/Server database.

➢ To run the ctmspdiag utility:

Type one of the following commands to invoke the ctmspdiag utility:

- `ctmspdiag -SET -SPNAME <storeProcedureName> -MODE <Y/N>`
- `ctmspdiag -PRINT -SPNAME <storeProcedureName>`
- `                 [-FROM_DATE<timeStamp>] [-TO_DATE <timeStamp>]`
- `ctmspdiag -DEL -TO_DATE <timeStamp>[-SPNAME <storeProcedureName>]`
- `ctmspdiag -SHOW [-SPNAME <storeProcedureName>]`
- `ctmspdiag -TRUNCATE`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmspdiag parameters, see ctmspdiag parameters (on page 395).

ctmspdiag options

The following table lists the options that the ctmspdiag utility supports:

Option	Description
-SET	Set the diagnostic request status of a specified SP in the Control-M/Server database. ▪ If the diagnostic request status does not exist in the Control-M/Server database, it is created and its value is set. ▪ If the diagnostic request status does exist, its value is updated.
-PRINT	Print the recorded diagnostic messages for the specified SP name. The SP name can include the wildcard character * to indicate any number of characters (including none). When using an *, enclose the SP name in quotation marks, for example, "clean* ".
-DEL	Erase diagnostic messages that are equal to, or older than, the TO_DATE parameter.
-TRUNCATE	Erase all SP diagnostic messages in the Control-M/Server database.
-SHOW	Displays the current diagnostic request status for the specified SP. If the SP name is omitted, the diagnostic request status of all the SPs is displayed.

ctmspdiag parameters

The following table lists the parameters that can be specified with the ctmspdiag options (on page 394):

Parameter	Description
-SPNAME	Specifies the stored procedure name.
-MODE	Specifies whether diagnostic messages from SPs should be recorded in the Control-M/Server database. The values are: ▪ Y ▪ N (Default)
-FROM_DATE	Start date for the date range of diagnostic messages to print. The date is in yyyymmddhhmm format. If both this parameter and TO_DATE are not specified, all the messages are printed.
-TO_DATE	End date for the date range of diagnostic messages to print. Each date is in yyyymmddhhmm format. ▪ For the -PRINT option, if both this parameter and -FROM_DATE are not specified, all the messages are printed. ▪ For the -DEL option, if this parameter is not specified, all the messages are deleted.

ctmspdiag examples

The following are the ctmspdiag utility commands:

This example sets the status request value Y for the SP named saturn in the Control-M/Server database.

```
ctmspdiag -SET -SPNAME saturn -MODE Y
```

This example prints messages found in the Control-M/Server database that were recorded from the SP named saturn.

To print the date range from minute after midnight on November 14, to midnight on December 20, 2007, specify the following command:

```
ctmspdiag -PRINT -SPNAME saturn -FROM_DATE 200711140001 -TO_DATE
200712200000
```

To print all the messages, specify the following command:

```
ctmspdiag -PRINT -SPNAME saturn
```

This example erases all the messages older than 08:30 December 14, 2007 found in the Control-M/Server database, that were recorded from the SP named saturn.

```
ctmspdiag -DEL -TO_DATE 200712140830 -SPNAME saturn
```

This example erases all the diagnostic information found in the Control-M/Server database, that were recorded from SPs.

```
ctmspdiag -TRUNCATE
```

This example displays the current diagnostic request status for the SP named saturn.

```
ctmspdiag -SHOW -SPNAME saturn
```

ctmsuspend

The ctmsuspend utility suspends and restores Control-M for Databases non-communication processes for mass uploads and downloads from Control-M/EM. During suspension mode, Control-M deactivates its job processing functions by suspending the TR, SL, NS, LG, and WD processes. To run the ctmsuspend utility, see Runningthe ctmsuspend utility (on page 396).

This utility should be invoked before executing the Mass Upload or Mass Download features on the Control-M Workload Automation.

Runningthe ctmsuspend utility

This procedure describes how to run the ctmsuspend utility, which enables you to suspends and restores Control-M for Databases non-communication processes for mass uploads and downloads from Control-M/EM.

➢ To run the ctmsuspend utility:

- Type the following command to invoke the ctmsuspend utility:

  ```
  ctmsuspend {-s|-r}
  ```

  -s: Suspends Control-M for Databases scheduling processes. Leaves the gateway to Control-M/EM open.

  -r: Restores Control-M for Databases processes. Resumes normal operating mode.

The following command causes Control-M for Databases scheduling processes to be restored.

```
.ctmsuspend -r
```

init_prflag

The init_prflag utility resets sleep times and trace levels for Control-M/Server processes. To run the init_prflag utility, see Running the init_prflag utility (on page 397).

This utility performs the following actions:

- The sleep times for all Control-M/Server processes are reset to their initial (installation) values. See the table below.
- The trace level for all Control-M/Server processes is reset to zero.

The sleep time setting for Control-M/Server processes can affect the functionality of Control-M/Server and the performance of your data center. Sleep time is the length of time that a process lies dormant before "waking up" to check if any request to perform an action was received. When modifying certain Control-M/Server process sleep time settings, it is important to consider the number of jobs that are processing, the job schedule plan, and the overall load on the computer.

Running the init_prflag utility

This procedure describes how to run the init_prflag utility, which enables you to reset sleep times and trace levels for Control-M/Server processes.

➢ To run the init_prflag utility:

1. Shut down Control-M/Server.
2. Type the following command at the command prompt:

```
init_prflag
```

init_prflag sleep time considerations

The following table lists the sleep time considerations for the init_prflag utility:

Process	Task	Sleep Time Initial Settings	Sleep TIme Modification Considerations
SU	Supervisor	60	Increase: Delay in startup, downloads, and New Day procedure.
RT	Inter-process Communication Router	30	No effect.
WD	Watchdog	360	No effect.
SL	Job Selector and job post processing	5	Increase: Delay in Job Submission and Shout messages for late submission. Can be increased during period of minimal job processing. Decrease: Additional CPU resources.
TR	Job Tracking	5	Increase: Delay in freeing resources after job ends and delay in Shout messages. Can be increased during period of minimal job processing. Decrease: Additional CPU resources.
NS	Communication with Control-M/EM	30	No effect.
LG	Utilities Agent invoker	360	No effect.
CO	Communication agents	60	No effect.
CD	New Day Procedure; Database uploads and downloads	60	No effect.
CS		N/A	No effect.
CA		N/A	No effect.

Running the shut_ca utility

This procedure describes the shut_ca utility, which enables you to to shut down the Control-M/Server Configuration Agent.

➢ To run the shut_ca utility:

1. Run the following command at the command prompt:

```
shut_ca
```

2. To shut down the Control-M/Server Configuration Agent, run the following command at the command prompt:

```
shut_ca
```

The following messages are displayed:

```
---------------------------------------------------
Shutting down Control-M/Server Configuration Agent
---------------------------------------------------
Waiting .....
Control-M/Server Configuration Agent is down
```

Running the shut_ctm utility

This procedure describes how to run the shut_ctm utility, which enables you to shut down Control-M/Server and its processes.

➢ To run the shut_ctm utility:

- To shut down Control-M/Server, issue the following command:

```
shut_ctm
```

Running the show_ca utility

This procedure describes how to run the show_ca utility, which enables you to display the status of the Control-M/Server Configuration Agent.

➢ To run the show_ca utility:

- To display the status of the Control-M/Server Configuration Agent, issue the following command:

```
show_ca
```

Running the shutdb utility

This procedure describes how to run the shutdb utility, which enables you to shut down the SQL database server.

The SQL database server can also be shut down using the Control-M/Server menu options.

Before you begin

- The SQL Database Server must be started before Control-M/Server is started and must be active as long as Control-M/Server is active. Mandatory.

➢ To shut down the SQL database server:

1. Log on as the Control-M/Server account owner.
2. Run the following command from the command prompt:

```
shutdb
```

Running the start_ca utility

This procedure describes how to run the start_ca utility, which enables you to start up the Control-M/Server Configuration Agent.

To run the start_ca utility:

- Do on of the following:
  - To start up the Control-M/Server Configuration Agent, run the following command:

    ```
    start_ca
    ```

  - Start up the Control-M/Server Configuration Agent by running the following command:

    ```
    start_ca
    ```

    A message similar to the following is displayed:

    ```
    Starting Control-M/Server Configuration Agent at Fri Jun 4 14:31:27 IDT
    2010
    ```

Running the start_ctm utility

This procedure describes how to run start_ctm utility, which enables you to start the Control-M/Server.

➢ To run the start_ctm utility:

- Start Control-M/Server by running the following command at the command prompt:

```
start_ctm
```

Running the startdb utility

This procedure describes how to run the startdb utility, which enables you to start the SQL database server.

The SQL database server can also be started using the Control-M/Server menu options.

Before you begin:

- The SQL Database Server must be started before Control-M/Server is started and must be active as long as Control-M/Server is active. Mandatory.

➢ To run the startdb utility:

1. Log on as the Control-M/Server account owner.
2. Run the following command from the command prompt:

```
startdb
```

Control-M/Agent

This table list the Control-M/Agent utilities for communication, startup, and troubleshooting.

Utility Type	**Description**
ag_diag_comm (on page 401)	BMC recommends that you verify the ability of the agent computer to communicate with the primary Server computer and with all other authorized Server host computers.
Running the ag_ping utility (on page 406)	This utility verifies that Control-M/Server is active on the Server computer connected to the agent computer.
Running the shagent utility (on page 406)	The shagent utility (*UNIX only*) checks that the p_ctmag and p_ctmat processes are running. It can be invoked only from the Control-M/Agent computer.

ag_diag_comm

BMC recommends that you verify the ability of the agent computer to communicate with the primary Server computer and with all other authorized Server host computers.

Control-M/Agent includes a diagnostic program that checks parameters and environmental conditions relevant to communication between the agent and server computers. This program is typically used at the request of Technical Support to determine the cause of a communication problem.

Running the ag_diag_comm utility

This procedure describes how to run the ag_diag_comm utility, which enables you to generate a diagnostic report on the Control-M/Agent communication.

➢ To run the utility:

1. Navigate to the directory in which Control-M/Agent is installed.
2. Enter the following command:

   ```
   ag_diag_comm
   ```

   The Control-M/Agent Communication Diagnostic Report is displayed. See check output below.

   If the user is not the administrator, the ag_diag_comm command cannot display all the details.

Modifying recovery settings for network disconnections

This procedure describes how to modify recovery settings for network disconnections.

➢ To modify recovery settings for network disconnections:

1. In the computer where Control-M/Agent is installed, modify the registry keys (for Windows), or modify the parameters in the **OS.dat** file (for UNIX), as needed.

   Use the following table for reference. The path to the registry keys is **HKEY_LOCAL_MACHINE\SOFTWARE\BMC Software\Control-M/Agent\WIN**, and the path to the **OS.dat** file is *agentInstallDir***/ctm/data**.

2. Restart the Control-M/Agent services on Windows, or the processes on UNIX, to make these changes take effect.

ag_diag_comm output example

The output of the utility is similar to the following:

```
Control-M/Agent Communication Diagnostic Report
-----------------------------------------------
Agent User Name                : ag620
Agent Directory                : /home/ag620/ctm
Agent Platform Architecture    : AIX
Agent Version                  : 6.3.01.000
Agent Host Name                : appsrv002
Server Host Name               : sunsrv001
Authorized Servers Host Names  : sunsrv001
Server-to-Agent Port Number    : 7006
Agent-to-Server Port Number    : 7005
Server-Agent Protocol Version  : 06
 Server-Agent Comm. Protocol    : TCP
Server-Agent Connection mode   : Transient
Unix Ping to Server Platform   : Succeeded
 Agent Ping to Control-M/Server : Succeeded
Agent processes status
======================
Agent Router                   : Not running
Agent Listener                 : Running (42762)
 Agent Tracker                  : Running (51208)
```

Network disconnections

Control-M/Agent requires an open connection to a remote host from the time of job submission until the end of the job. If a network disconnection occurs, Control-M/Agent attempts to restore the connection. During the reconnection attempts, jobs running on the remote host, through the specified Control-M/Agent, remain in executing status.

If the connection is restored, the status of jobs is updated to reflect their current status, either completed or still running. If the connection is not restored after the specified number of attempts, the jobs end with **NOTOK** status.

The handling of network disconnections to remote hosts is supported when you use remote hosts on UNIX, Linux, and z/OS USS (using SSH) and when you use remote hosts on Windows (using WMI).

The OUTPUT and exit code for jobs are stored in files on the remote host. The files reside in the user home directory of the job’s owner, in the directory specified by RJX_OUTPUT_DIR. (For more information about RJX_OUTPUT_DIR, see the table below.) If network connections were restored, these files are deleted when the jobs end. If network connections were not restored, you can check these files to see if the jobs completed successfully or failed.

The following section describes how to use configuration parameters to modify recovery settings when a network disconnection occurs.

Network disconnection parameters and registry keys

The following table lists the ag_diag_comm utility network disconnection parameters and registry keys.

Parameter name (UNIX) or registry key (Windows)	Description
RJX_CONN_TRY	number of attempts made to restore the connection The default is **15**
RJX_CONN_TOUT	time interval between attempts to restore the connection The default is **120 seconds**
RJX_OUTPUT_DIR	directory to store temporary files These files are automatically removed to Control-M/Agent when the jobs end and the network connection is available or restored. The default is blank. Blank means that the files are stored in the user home directory of the job's owner in the remote host.
RJX_KEEP_OUTPUT	to prevent deletion of the OUTPUT file from the remote host
RJX_COPY_OUTPUT_REMOTE	to determine where the OUTPUT handling operations of copy, move, or delete are performed, either on the Agent or the remote host
RJX_CONN_RECONNECT	whether the network reconnection feature is enabled The default is **Y**
RJX_CONN_SFTP	protocol used to upload and download files: When **Y** is specified, the agent uses SFTP (secured FTP) protocol to retrieve, upload, and download files. When **N** is specified, the agent uses the SSH protocol to retrieve, upload, and download files. The default is **Y**

Running the ag_ping utility

This procedure describes how to run the ag_ping utility, which enables you to verify that Control-M/Server is active on the Server computer connected to the agent computer.

➢ To run the ag_ping utility:

- From the operating system prompt, specify the following command:

  ```
  ag_ping
  ```

  The utility attempts to communicate with Control-M/Server and indicates whether the attempt succeeded or failed.

  If the attempt succeeds, a message similar to the following is displayed:

  ```
  Output:
  Server is alive.
  Result: Success.
  ```

Running the shagent utility

This procedure describes how to run the shagent utility (*UNIX only*), which enables you to check that the p_ctmag and p_ctmat processes are running. It can be invoked only from the Control-M/Agent computer.

➢ **To run the shagent utility:**

1. From the operating system prompt, specify the following command:

   ```
   shagent
   ```

   The shagent utility has no parameters.

   If the ctmag advanced utility parameter Persistent Connection is set to **Y**, the utility verifies that the p_ctmar process is running.

2. To check that the p_ctmag and p_ctmat processes are running, specify the following command:

   ```
       shagent
   ```

   If the Router process is running, output similar to the following is displayed:

   ```
       root         7660   0:00 p_ctmag
       root         7644   0:00 p_ctmar
       root         7745   0:29 p_ctmat
   ```

Administration and configuration

The administration and configuration utilities perform various administration and configuration tasks.

By including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utilities used for administration and configuration tasks

Utility	Description
agkeystore (on page 453)	Create, apply, and restore keys for job owners credentials using a Blowfish algorithm.
ccmcli (on page 409)	Performs basic management operations and delete old alerts.
Health Check (on page 462)	Scans and collects system information about the environment.
ctm_menu (on page 421)	Interactive menu that provides access to a variety of functions and utilities that are used to maintain Control-M/Server.
ctmag (on page 458)	Agent Configuration utility that is used to maintain Control-M/Agent configuration parameters, and to view and modify most of the operating system parameters.
Running the ctmagcfg utility (on page 458)	Interactively configures Control-M/Agents.
ctmagcln (on page 423)	Sends requests to all available Control-M/Agents, or to a specified Control-M/Agent, to remove all OUTPUT files and status files that are no longer needed.
ctmdiskspace (on page 424)	Checks the amount of free disk space on a device.
ctmkeygen (on page 426)	Generates SSH private and public key pairs.
ctmkeystore_mng (on page 431)	Creates, modifies, and deletes keys with secure information such as passwords.

Utility	Description
ctmldnrs (on page 436)	Creates and loads the Manual Conditions file.
ctmlog (on page 442)	Performs a selective cleanup of the Control-M log or produces a report of Control-M log entries.
ctmsys (on page 445)	Maintains Control-M/Server system parameters and Shout Destination tables.
Running the ctmunixcfg utility (on page 458)	Enables you to view and modify most of the configuration parameters in the **OS.dat** file.
Running the ctmwincfg utility (on page 452)	Enables you to view and modify the Application Add-on for Windows configuration parameters.
ecaqrtab (on page 416)	Performs operations on quantitative resources in the Resources table.
erase_audit_data (on page 413)	Manually cleans up audit information from the Control-M/EM database.
purge_runinfo (on page 477)	Performs manual cleanup of run information that is retained by Control-M/Forecast and Control-M Batch Impact Manager for the purposes of performing its calculations.
purge_xalerts (on page 415)	Deletes exception alerts from the Exception Alerts table in the Control-M/EM database.
set_agent_mode (on page 459)	Handles the necessary changes to permissions of several agent files and directories in order to allow it to run jobs with any owner in non-root mode or revert back to the original state.

Control-M/EM utilities

This table lists the Control-M/EM utilities for administration and configuration.

Utility Type	Description
ccmcli (on page 409)	The ccmcli utility enables you to interactively perform basic administrative tasks on Control-M components using the Control-M Configuration Manager GUI
erase_audit_data (on page 413)	The erase_audit_data utility deletes records written prior to the specified date.
purge_xalerts (on page 415)	The purge_xalerts utility deletes exception alerts from the Exception Alerts table in the Control-M/EM database.
ecaqrtab (on page 416)	The ecaqrtab utility performs operations on the Quantitative Resources table.

ccmcli

The ccmcli utility enables you to interactively perform basic administrative tasks on Control-M components using the Control-M Configuration Manager GUI, including the following tasks:

- Starting
- Stopping
- Ignoring
- Recycling
- Viewing details about the component or server
- Removing old alerts
- Deleting history of job or group definitions
- Deleting Control-M/Agent or remote host component

To run the ccmcli utility, see Running the ccmcli utility (on page 410).

Running the ccmcli utility

This procedure describes how to run the ccmcli utility, which enables you to perform basic administrative tasks on Control-M components.

➢ To run the ccmcli utility:

- Type the following command:

  -
    ```
    em ccmcli [-u <user> [-p <password>] | -pf <passwordFile>]
      [-s <Configuration Server Name>] [-t <component Type>]
      [-n <component Name>] [-h <component Host>] [-cmd <command>]|
      [-date <YYYYMMDD>] | [-keep_days <number>] | [-force Y|N] | [-node_id
    <Node ID>] | [-ctlcmd <command>] ]
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For the ccmcli parameters, see ccmcli parameters (on page 411).

ccmcli parameters

The following table lists the ccmcli utility parameters:

Parameter	Description
-u	Control-M for Databases user name
-p	Control-M for Databases user password

-pf	Flat file containing an unencrypted username and password on separate lines in the format:
	user=*username*
	password=*password*

-s	The name of the Control-M Configuration Server. It is mandatory to specify this parameter.
-t	The type of component. Valid values are: ▪ Gateway ▪ GUI_Server ▪ GCS ▪ BIM ▪ Forecast_Server ▪ Self_Service_Server ▪ Web_Server ▪ CTM_Server ▪ CTM_Agent
-n	The logical name of the component.
-h	The host on which the component is running.

Parameter	Description
-cmd	The command you want to run on the agent configuration server or component. Valid values are: ▪ start ▪ stop ▪ ignore ▪ recycle ▪ details ▪ remove_old_alerts ▪ erase_jobs_history ▪ delete_agent ▪ delete_remote_host If the component type is **-t CTM_Agent**, only **-cmd recycle** is supported. When you specify the -force flag, and the recycle command for a Control-M/Agent, you can force recycling of a Control-M/Agent, even if jobs are running on it or on a remote host computer. You must specify the -date flag, when you specify the remove_old_alerts command. You must specify the -node_id flag, when you specify the delete_agent and delete_remote_host commands. You must specify the Control-M/Server where the Control-M/Agent and Remote Host are defined, when you specify the command type, name , and host for the delete_agent and delete_remote_host commands. You must specify the -keep_days flag, when you specify the erase_jobs_history command. All jobs above the specified number are deleted.
-date	The date from which alerts must be kept. Alerts older than the specified date are removed. Enter the date in the format YYYYMMDD. Use this parameter when specifying **-cmd remove_old_alerts**. All alerts received up to and including the specified date are deleted.
-keep_days	The number of days to keep job or group definition history. Use this parameter when specifying **-cmd erase_jobs_history**. All versions older than the specified number of days will be deleted.

Parameter	Description
-force	The command to force the component to recycle. Valid values are: ▪ Y ▪ N When **-cmd recycle** is specified for an agent (**-t CTM_Agent**), you can specify the **-force** to force Control-M/Agent to recycle even if jobs are running on the agent or remote host computer through the agent.
-node_id	The name of a Control-M/Agent computer, remote host computer, or node group where the job is submitted.
-ctlcmd	A control shell command sent to one of the Control-M/EM components.

ccmcli sample

The following are example commands of the ccmcli utility:

Use the following command to perform the administrative task of starting Control-M/Server on the alpine host computer from the command prompt.

```
em ccmcli -pf dailyuser -s alpha -t CTM_Server -n alpha_server -h
alpine -cmd start
```

The following command keeps the last 15 days of job definition history for user user014 and deletes the older versions of the job definition history. The output of the utility shows the number of versions before and after the deletion. However, jobs can be running while the utility is executing so the difference between them is not necessarily the number of versions that were deleted.

```
em ccmcli -u user014 -p pass104 -cmd erase_jobs_history -keep_days 15
-s server04
```

Utility output:

Number of old versions before deletion: 8512.

Number of old versions after deletion: 3478.

erase_audit_data

The erase_audit_data utility deletes records written prior to the specified date.

When erase_audit_data is invoked, it uses a script to delete records written before a specified date. If the -U and -P parameters are not specified, the Control-M/EM database owner (DBO) user name and password are prompted for. The erase_audit_data utility can delete large numbers of audit records. To run the erase_audit_data utility, see: Running the erase_audit_data utility (on page 414).

Running the erase_audit_data utility

This procedure describes how to run the erase_audit_data utility, which deletes records written prior to the specified date.

➢ To invoke the exportdefjob utility:

1. Do one of the following:
   - **a.** Log on to a Control-M/EM account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

   ```
   erase_audit_data [-date YYYYMMDD] [-U emDboName]
   [-P emDboPassword]
   ```

   Records written prior to the specified date are deleted.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail on the erase_audit_data utility, see: erase_audit_data utility parameters (on page 414).

erase_audit_data utility parameters

The following table describes the erase_audit_data parameters:

Parameter	Description
-date	A date in YYYYMMDD format (for example, 20070915), which sets the selection criteria for obsolete jobs. Default: two days before the current date
-U	Name of the Control-M/EM database.
-P	Password of the Control-M/EM database.

When cleanup of audit information from the Control-M/EM database is automatic, the MaxAuditsToDelete parameter specifies the maximum number of audit records to delete during each automatic cleanup operation. If the number of audit records to clean is higher than this number, no records are deleted. The default is 400000 records. If the number of audit records to clean is higher than the MaxAuditsToDelete parameter, a message is issued to the GUI Server diagnostic log asking you to clean audit records manually using the erase_audit_data utility. You can use the Control-M/EM Administration facility to change the automatic cleanup of audit information settings.

purge_xalerts

The purge_xalerts utility deletes exception alerts from the Exception Alerts table in the Control-M/EM database. To run the purge_xalerts utility, see: Running the purge_xalerts utility (on page 415).

The deleted exception alerts are no longer displayed in the XAlerts window either within an hour or after you restart the CMS, whichever occurs first.

For more information about exception alerts, see Managing exception alerts.

Running the purge_xalerts utility

This procedure describes the purge_xalerts utility, which deletes exception alerts from the Exception Alerts table in the Control-M/EM database.

➢ To invoke the utility:

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

   ```
   purge_xalerts [-U <emDBO>] [-P <emDBOPassword>]  [-keep_days <number>]
   ```

   The utility has finished when the message Ended successfully is displayed.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail on the purge_xalerts utility, see: purge_xalerts utility parameters (on page 415).

purge_xalerts utility parameters

The following table describes the purge_xalerts utility parameters:

Parameter	Description
-U	Control-M/EM database user name.
-P	Control-M for Databases database user password.
keep_days	The number of days for which exception alerts are kept in the Control-M/EM database.

ecaqrtab

The ecaqrtab utility performs operations on the Quantitative Resources table. These operations include:

- Listing quantitative resources. To list quantitative resources, see: Listing quantitative resources (on page 418)
- Adding/deleting a quantitative resource. To add and delete quantitative resources, see: Adding a Quantitative resource (on page 419) and Deleting a quantitative resource (on page 419)
- Manually changing availability of a quantitative resource. To manually change the availability of a quantitative resource, see: Alerting the availability of a quantitative resource (on page 420)

To run the ecaqrtab, see: Running the ecaqrtab utility (on page 416)

Running the ecaqrtab utility

This procedure describes how to run the ecaqrtab utility, which performs operations on the Quantitative Resources table.

➢ To run the ecaqrtab utility:

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Do the following:

- Enter one of the following commands to invoke the ecaqrtab utility from the server:
  - o `ecaqrtab {LIST|ADD|DELETE|UPDATE}{<orName>|"*"}[<Max>][-OUTPUT <output>]`
  - o `ecaqrtab -input_file <fullPathFileName>`
- Enter one of the following commands to invoke the LIST option of this utility from the agent:
  - o `ecaqrtab LIST "*" [-OUTPUT <output>]`
  - o `ecaqrtab -input_file <fullPathFileName>`

  If a resource name is longer than 20 characters, the resource is not added.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information on the ecaqrtab utility, see: ecaqrtab utility parameters (on page 417).

ecaqrtab utility parameters

The following table describes the ecaqrtab utility parameters:

Parameter	Description
LIST	Displays the status of the specified Quantitative resources. This information is also available from Control-M/EM in the Quantitative Resources window.
ADD	Defines a new Quantitative resource and sets the maximum availability for the resource.
DELETE	Deletes an existing Quantitative resource.
UPDATE	Changes the maximum availability of an existing Quantitative resource.
<QR_Name>	Name of the Quantitative resource. For the LIST option, QR_Name can include wildcard character * to indicate any number of characters (including none). If * is specified, enclose the name in quotation marks, for example, "LVL*". When invoked by the server or agent, specify "*" (including the quotation marks) to include all existing Quantitative resources (default when invoked by the server).
<Max>	Maximum availability for the specified resource. Can only be specified with the ADD and UPDATE options.
<Output>	Full path name to which the report should be sent (optional). If not specified, output is routed to the default output device. This parameter can only be specified with the LIST option.
<fullPathFile Name>	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: Prepare and save files of utility parameters that can be reused. Specify utility input longer than the number of characters allowed in the command line. `-input_file ~<controlmOwner>/ctm_server/data/ecaqrtab_parms.txt` The path that is specified for this parameter must be accessible by the Control-M/Server account (even if this utility is requested by Control-M/Agent.

Listing quantitative resources

This procedure describes how to list the quantitative resources.

➢ To list quantitative resources:

1. Do one of the following:
   a. Log on to a Control-M/EM account (*UNIX*)
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Do one of the following:
   - Enter the following command when invoked by the server:
     ```
     o  ecaqrtab LIST {<QR_Name>|"*"} [-OUTPUT <Output>]
     ```
   - Enter the following command when invoked by the agent:
     ```
     o  ecaqrtab LIST "*" [-OUTPUT <Output>]
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details, see Quantitative Resource parameters (on page 418).

Quantitative Resource parameters

The following table describes the fields that are displayed for each Quantitative resource that matches the specified resource name or mask.

Parameter	Description
QR name	Quantitative resource name (with @<Host ID> where applicable).
Type	For future use.
Max Avail	Maximum number of units of this resource in the computer.
Reserved	Number of units of the resource reserved for critical-path jobs.
Used	Number of units of the resource currently in use or reserved. If the ctmloadset utility is used in the data center, this number can include usage of the resource by non-Control-M jobs.
Free	Number of units of the resource currently available for use. This represents the difference between Max Avail and Used.

Quantitative Resource example

The following command can be invoked by the server or the agent to list the current status of all Quantitative resources in the Quantitative Resource table:

```
ecaqrtab LIST "*" -OUTPUT D:\ctm_server\QR.txt
```

A report similar to the following is displayed:

```
+--------------------------------------------------------------+

Resource Name  Type  Max-Avail  Reserved  Used  Free

+--------------------------------------------------------------+

CPU@linda       L        10          0       10     0

CPU@linda       L        20          0       15     5

MEM@diana       L        10          0        0    10

Tape2           L        12          2        2    10
```

Adding a Quantitative resource

This procedure describes how to add a Quantitative Resource.

➢ To add a quantitative resource:

1. Do one of the following:

a. Log on to a Control-M/EM account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
ecaqrtab ADD <QR name> <Max>
```

The following command specifies that the new Quantitative resource tape2 is to be added to the Quantitative Resource table, with a maximum availability of 12 units:

```
ecaqrtab ADD tape2 12
```

Deleting a quantitative resource

This procedure describes how to delete a quantitative resource.

➢ To delete a quantitative resource:

1. Do one of the following:

a. Log on to a Control-M/EM account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
ecaqrtab DELETE <QR name>
```

The following command specifies that the Quantitative resource tape3 is to be deleted from the table:

```
ecaqrtab DELETE tape3
```

Alerting the availability of a quantitative resource

This procedure describes how to alter the availability of a quantitative resource.

➢ To alter the availability of a Quantitative resource:

1. Do one of the following:
   - **a.** Log on to a Control-M/EM account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

```
ecaqrtab UPDATE <QR name> <Max>
```

The following command specifies that the new maximum availability for the existing Quantitative resource linerje2 on computer diana is 12 units:

```
ecaqrtab UPDATE linerje2@diana 12
```

You can get the same result using the -input_file parameter as follows:

```
ecaqrtab -input_file ~<controlm_owner>/ctm_server/data/ecaqrtab_lines.txt
```

The referenced file contains the following lines:

```
UPDATE

linerje2@diana

12
```

e.

Control-M/Server utilities

This table lists the Control-M/Server utilities for administration and configuration.

Utility Type	Description
ctm_menu (on page 421)	ctm_menu (Control-M Main Menu) is a utility that invokes an interactive menu that provides access to a variety of functions and utilities that are used to maintain Control-M/Server.
ctmagcln (on page 423)	The ctmagcln utility sends a request to all available Control-M/Agents, or to a specified Control-M/Agent, to remove all OUTPUT files and exit status files that are no longer needed.
ctmdiskspace (on page 424)	The ctmdiskspace utility checks the amount of free disk space on a device and displays the result.
ctmkeygen (on page 426)	The ctmkeygen utility generates SSH private and public key pairs.
ctmkeystore_mng (on page 431)	The ctmkeystore_mng utility enables you to do the following actions: ▪ For Remedy – create, modify, and delete user names and passwords. ▪ For Blowfish – add, apply, and restore encrypted keys for user authorizations (job owners credentials).
ctmldnrs (on page 436)	The ctmldnrs utility creates and loads the Manual Conditions file.
ctmlog (on page 442)	The ctmlog utility creates a report from entries in the Control-M log or deletes entries in the Control-M log.
ctmsys (on page 445)	The ctmsys utility is an interactive utility for maintaining: ▪ Shout Destination tables (for directing Shout messages). ▪ Control-M system parameters.

ctm_menu

ctm_menu (Control-M Main Menu) is a utility that invokes an interactive menu that provides access to a variety of functions and utilities that are used to maintain Control-M/Server. To run the ctm_menu utility, see Running the ctm_menu utility (on page 422).

Running the ctm_menu utility

This procedure describes how to run the ctm_menu utility, which invokes an interactive menu that provides access to a variety of functions and utilities that are used to maintain Control-M/Server.

➢ To run the ctm_menu utility:

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

   ```
   ctm_menu
   ```

   The following interactive menu is displayed:

   ```
   Control-M Main Menu
   --------------------------------------------------§
   §
   Select one of the following menus:§
   §
   1 - CONTROL-M Manager§
   2 - Database Creation§
   3 - Database Maintenance§
   4 - Database Mirroring§
   5 - Security Authorization§
   6 - Parameter Customization§
   7 - Node Group§
   8 - View Node ID details§
   9 - Agent Status§
   10 - Troubleshooting§
   §
   q - Quit§
   ```

3. To use the Control-M Main Menu and its submenus:

   a. Enter the number of the menu option.

   b. Make the required changes.

   c. When you are done, enter **q** to quit.

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

ctmagcln

The ctmagcln utility sends a request to all available Control-M/Agents, or to a specified Control-M/Agent, to remove all OUTPUT files and exit status files that are no longer needed. This utility should run while Control-M/Server is up and running. The files that are no longer needed are determined according to the Maximum Days to Retain Output Files parameter. For more information, see the description of the Output parameters parameter. To run the ctmagcln utility, see Running the ctmagcln utility (on page 423).

The New Day procedure can request that Agents remove OUTPUT files and exit status files that are no longer needed. In an environment with multiple Agents, it can take a long time to send the cleanup request to the Agents during New Day. BMC recommends that you use the AGENTS_CLEANUP_IN_NEWDAY configuration parameter to disable this action during the New Day procedure, and instead use the ctmagcln utility. For more information, see the description of the Configuration parameters configuration parameter.

The ctmagcln utility can be run as a daily job. A message is added to the IOALOG when the cleanup of the Agents has ended.

Running the ctmagcln utility

This procedure describes how to run the ctmagcIn utility, which ends a request to all available Control-M/Agents, or to a specified Control-M/Agent, to remove all OUTPUT files and exit status files that are no longer needed

➢ To run the ctmagcIn utility:

1. Do one of the following:
   - **a.** Log on to a Control-M/EM account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

```
ctmagcln -agent <"*" | agentName> [-days <outputRetainDays>]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail on the ctmagcIn utility, see ctmagcln utility parameters (on page 424) and ctmagcIn utility example (on page 424).

ctmagcln utility parameters

The following table describes the parameters of this command:

Parameter	Description
*	Specifies that all Agents remove OUTPUT files and exit status files that are no longer needed.
agentName	Specifies that the specified agent must remove OUTPUT files and exit status files that are no longer needed.
outputRetainDays	Specifies the number of days that OUTPUT files must be retained before being removed. Default: **1**

BMC recommends running ctmagcln as soon as possible after the New Day procedure is complete.

ctmagcIn utility example

The following are examples of the ctmagcIN utility commands:

The following command requests that all Agents remove OUTPUT files and exit status files that are no longer needed.

```
ctmagcln -agent "*"
```

The following command requests that the specified agent, saturn, removes OUTPUT files that have already been retained for two days, and removes all exit status files that are no longer needed.

```
ctmagcln -agent saturn -days 2
```

ctmdiskspace

The ctmdiskspace utility checks the amount of free disk space on a device and displays the result. The utility returns a "failed" status if the current free space is below the specified limit.

The ctmdiskspace utility can be included in the Control-M Watchdog process. For more information, see Watchdog process parameters. To run the ctmdiskspace utility, see: Running the ctmdiskspace utility (on page 424).

Running the ctmdiskspace utility

This procedure describes how to run the ctmdiskspace utility, which checks the amount of free disk space on a device and displays the result.

➢ To run the ctmagcIn utility:

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<*EM Home*>\Default\bin** directory.

2. Enter the following command:

```
ctmdiskspace -limit <amount>{%|K|M} -path <pathName> [-quiet]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmdiskspace utility, see: ctmdiskspace parameters (on page 425) and ctmdiskspace utility example (on page 425).

ctmdiskspace parameters

The following table describes the parameters of this command:

Parameter	Description
Amount	Specifies the minimum amount (%, K, or M) of free space on the device as a whole number (integer). For example: -limit 25%
path_name	Specifies the full path name of the device. Multiple devices can be specified on the command line (see the second example below).
-quiet	Suppresses informational messages from being displayed during the execution of the command.

More than one **-path <***pathName***>** statement can be specified for each run of the ctmdiskspace utility.

ctmdiskspace utility example

The following are ctmdiskspace utility examples:

The following command returns a "failed" status if the amount of free disk space in the Control-M user directory is below 25%:

```
ctmdiskspace -limit 25% -path /ctm_server/ctmuser
```

The following command returns a "failed" status if the amount of free disk in the Control-M user directory is below 20M:

```
ctmdiskspace -limit 20M -path /ctm_server/ctmuser -path /ctm/tmp
```

ctmkeygen

The ctmkeygen utility generates SSH private and public key pairs. To run the ctmkeygen utility, see: Running the ctmkeygen utility (on page 426).

When creating or modifying the job owner definition, you can choose to use either public or private key authentication instead of password authentication. For more information about using the Control-M Configuration Manager, see Introduction to Control-M Configuration Manager.

The ctmkeygen utility manages the key table that contains the logical key name as the unique table key, the private key, and the key passphrase (encrypted). The generated public key (unencrypted) is stored in a file.

Running the ctmkeygen utility

This procedure describes how to run the ctmkeygen utility, which generate SSH private and public key pairs. The ctmkeygen utility can be run either in interactive mode or batch invocation.

➢ To run the ctmkeygen utility:

1. Do one of the following:
   - **a.** Log on to a Control-M/EM account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter one of the following commands:
   - **Interactive mode**:

     ```
     ctmkeygen
     ```

     The Control-M Key Generator Utility menu is displayed. The options in this menu and in all other menus provided by this utility can be selected by typing the option number or command letter and pressing **<Enter>**.
   - **Batch mode:** Specify one of the following commands:
     - `ctmkeygen -action add -name <logicalKeyName> -passphrase <keyPassphrase> -type rsa|dsa -bits 512|768|1024|2048|3076 -format openssh|ssh2 -path <publicKeyPath>`
     - `ctmkeygen -action update -name <logicalKeyName> -passphrase <keyPassphrase> [-type rsa|dsa] [-bits 512|768|1024|2048|3076] [-format openssh|ssh2] -path <publicKeyPath>`
     - `ctmkeygen -action delete -name <logicalKeyName> -passphrase <keyPassphrase>`
     - `ctmkeygen -action list`
     - `ctmkeygen -action export -filename <exportFileName>`
     - `ctmkeygen -action import -filename <importFileName> -data append|truncate`
     - `ctmkeygen help`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmkeygen utility see: ctmkeygen utility actions (on page 427), ctmkeygen utility parameters (on page 428) and Copying public keys to the SSH server (on page 429).

ctmkeygen utility actions

The following table describes the actions in the ctmkeygen utility:

Action	Description
add	Creates a new entry in the key table. It also verifies that a key with the same name does not exist. All the parameters are mandatory.
update	Modifies the details of an existing entry in the key table. The entry includes the same fields as used to create a new key pair. The updated entry replaces the existing entry in the key table in the database and the public key file. The passphrase must match the one that was used to create the existing key. For the optional parameters, if a value not specified, the value stored in the Control-M/Server database is used.
delete	Deletes the entry associated with the logical key name. The passphrase must match the one that was used to create the existing key.
list	Returns a list of lines, each containing: the logical key name, type, bits, and format.
export	Exports the details of the keys stored in the key table to a text file. `ctmkeygen -action export -filename $HOME/ctm_server/data/key_details.txt`
import	Imports the details of the keys stored in the key table. Using the import parameter enables you to: prepare and save files of keys that can be reused specify utility input longer than the number of characters allowed in the command line `ctmkeygen -action import -filename $HOME/ctm_server/data/key_details.txt`
help	Displays the usage of the ctmkeygen utility.

ctmkeygen utility parameters

The following table describes the parameters in the ctmkeygen utility:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-name</td><td colspan="2">Logical name of the key that is used as a unique identifier. It also determines the name of the public key file. The name is comprised of letters, numbers, and underscores.</td></tr>
<tr><td>-passphrase</td><td colspan="2">Phrase used as a key to encrypt the key itself.</td></tr>
<tr><td>-type</td><td colspan="2">Specifies the standard used for the key. Mandatory when used with add, optional when used with update.Valid values are:
▪ RSA
▪ DSA</td></tr>
<tr><td>-bits</td><td colspan="2">Specifies the strength of the encryption key in bits. Mandatory when used with add, optional when used with update. Valid values are:
▪ 512
▪ 768
▪ 1024
▪ 2048
▪ 3076
The minimum value of the bits must be at least equal to the minimum value of bits specified for the SSH server.</td></tr>
<tr><td>-format</td><td colspan="2">Specifies the public key file format. It must match the format used by the SSH server. Mandatory when used with add, optional when used with update. Valid values are:
▪ openssh – for OpenSSH servers
▪ ssh2 – for ssh2 servers</td></tr>
<tr><td>-path</td><td colspan="2">Specifies the location where the public key file is created.</td></tr>
<tr><td>-filename</td><td colspan="2">Specifies the public key name. The format of the file depends on what is specified for the –format parameter, described above.</td></tr>
<tr><td rowspan="2">-data</td><td colspan="2">Describes what action to take with the imported data from the text file. Specify one of the following:</td></tr>
<tr><td>append</td><td>the details of the SSH keys from the imported text file are added to the existing SSH keys</td></tr>
</table>

Parameter	Description	
	truncate	the details of the SSH keys from the imported text file replace the existing SSH keys

Copying public keys to the SSH server

This procedure describes how to copy public keys to the SSH server. The public key must be copied to the SSH server. If such a file already exists on the SSH server, you must choose to either append or truncate the new file to the existing one.

➢ To copy public keys to the SSH server:

- Copy the public key to the SSH server according to the SSH server requirements:
  - For OpenSSH on UNIX, the public keys file is:
  - *<jobOwnerHomeDirectory>***/.ssh/authorized_keys**
  - For SSH Tectia on UNIX, the public keys file is:
  - *<jobOwnerHomeDirectory>***/.ssh2/authorization**
  - For SSH Tectia on WINDOWS, the public keys file is:

  *<jobOwnerHomeDirectory>***\.ssh2\authorization**

Copy public keys to SSH server example

Create an entry in the key table with the following specifications:

Parameter	Value
key name	key1
Passphrase	myphrase
Type	dsa
Bits	512
Format	ssh2
Path	/home/ctm630

Specify the following command:

```
ctmkeygen -action add -name key1 -passphrase myphrase -type dsa -bits 512
-format ssh2 -path /home/ctm630
```

The following message is displayed:

```
Creating SSH key. Please wait...

SSH key created successfully.
```

Assume that modifications are required to the key created in Example 1. To change the type to rsa, the number of bits to 1024 and the format to openssh, specify the following command:

```
ctmkeygen -action update -name key1 -passphrase myphrase -type rsa -bits 1024
-format openssh -path /home/ctm630
```

The following message is displayed:

Updating SSH key. Please wait...

SSH key update ended successfully.

To delete the key entry created in Example 1, specify the following command:

```
ctmkeygen -action delete -name key1 -passphrase myphrase
```

The following message is displayed:

```
Entry deleted successfully.
```

To display a list of SSH keys in the key table, specify the following command:

```
ctmkeygen -action list
```

The following is displayed:

```
Name                           Type       Bits       Format

----                           ----       ----       ------

first                          RSA        512        OPENSSH
```

```
mykey                          RSA        1024       OPENSSH

2 keys were found.
```

To create an export text file containing the details of the SSH keys, specify the following command:

```
ctmkeygen -action export -filename /home/ctm630/my.exp
```

The following is displayed:

```
Exporting data, please wait...

Export ended successfully.

Check report file
~<controlm_owner>/ctm_server/proclog/export_report_5020.txt' for details.
```

To import the **my.exp** text file, which contains the details of the SSH keys that replaces the current information, specify the following command:

```
ctmkeygen -action import -filename /home/ctm630oe/my.exp -data truncate
```

The following message is displayed:

```
Importing data, please wait...

Import ended successfully.

Check report file
~<controlm_owner>/ctm_server/proclog/import_report_535a.txt' for details.
```

ctmkeystore_mng

The ctmkeystore_mng utility enables you to do the following actions:

- For Remedy – create, modify, and delete user names and passwords.
- For Blowfish – add, apply, and restore encrypted keys for user authorizations (job owners credentials).

To work with Do Remedy, first activate this utility.

To run the ctmkeystpre_mng utility, see: Running the ctmkeystore_mng utility (on page 431).

The options in this menu and in all other menus provided by this utility can be selected by typing the option number or command letter and pressing <**Enter**>.

Running the ctmkeystore_mng utility

This procedure describes how to run the ctmkeystore_mng utility, which enables you to do various actions in Remedy and Blowfish.

➢ To run the ctmskeystore_mng utiltiy:

1. Do one of the following:

  a. Log on to a Control-M/EM account (*UNIX*)

  a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

  b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
ctmkeystore_mng
The following menu is displayed:
+------------------------------------------------------+
|     Control-M/Server Keystore Management Utility     |
|                   Main Menu                          |
+------------------------------------------------------+
1)   REMEDY Keystore
2)   BLOWFISH Keystore
q)   Quit
Enter option:
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

REMEDY Keystore

Specify the ctmkeystore_mng utility to create, modify, and delete Remedy user names and passwords.

To access the Remedy Keystore menu, see: Running the REMEDY Keystore menu (on page 432). To create, delete or modify the user and passwords, see the following:

- Creating a REMEDY user and password (on page 433)
- Deleting a Remedy user (on page 433)
- Modifying the password of a REMEDY user (on page 434)

Running the REMEDY Keystore menu

This procedure describes how to access the REMEDY Keystore menu.

➢ To run the REMEDY Keystore Menu:

1. Follow the steps referred to in Running the ctmkeystore_mng utility (on page 431).
2. Select option **1** from the Control-M/Server Keystore Management Utility Main Menu.

   The following menu is displayed:

```
+--------------------------------------+
 REMEDY Keystore Menu
+--------------------------------------+
1)   Add User
2)   Delete User
3)   Update User password
q)   Quit
Enter option:
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information on the REMEDY Keystore utility, see: REMEDY Keystore Menu parameters (on page 433).

REMEDY Keystore Menu parameters

The following table lists the REMEDY keystore Menu parameters"

Parameter	Description
user	Specifies a user name. Specify an alphanumeric value from 1 through 127 characters long.
password	Specifies the password for the user defined above.

Creating a REMEDY user and password

This procedure describes how to create a REMEDY user and password.

➢ To create a REMEDY user and password:

1. Follow the steps referred to in Running the ctmkeystore_mng utility (on page 431).
2. Select Option **1** from the REMEDY Keystore Menu.

   You are prompted to specify a username and password.

   After confirming the password, a message is displayed stating that the user name was successfully added.
3. Press **Enter** to continue.

   The REMEDY Keystore Menu is again displayed.

Deleting a Remedy user

This procedure describes how to to delete a REMEDY user.

➢ To delete a REMEDY user:

1. Follow the steps referred to in Running the ctmkeystore_mng utility (on page 431).
2. Select Option **2** from the REMEDY Keystore Menu.

   You are prompted to specify a username.
3. Press **Enter** to continue.

   A message is displayed stating that the user name was successfully deleted.
4. Press **Enter** to continue.

   The REMEDY Keystore Menu is again displayed.

Modifying the password of a REMEDY user

This procedure describes how to modify the password of a REMEDY user.

➢ To modify the password of a REMEDY user:

1. Follow the steps referred to in Running the ctmkeystore_mng utility (on page 431).
2. Select Option **3** from the REMEDY Keystore Menu.

   You are prompted to specify a username.
3. Press **Enter** to continue.

   You are prompted to enter new password and then confirm it.
4. Press **Enter** to continue.

   The REMEDY Keystore Menu is again displayed

BLOWFISH Keystore

Specify the ctmkeystore_mng utility to add, apply, and restore encrypted keys for user authorizations (job owners credentials). For information about changes to the Blowfish key in Control-M/Agent, see agkeystore (on page 453). To access the BLOWFISH keystore menu, see: Running the BLOWFISH Keystore Menu (on page 434).

Running the BLOWFISH Keystore Menu

This procedure describes how to access the BLOWFISH Keystore menu.

➢ To run the BLOWFISH Keystore Menu:

1. Follow the steps referred to in Running the ctmkeystore_mng utility (on page 431)
2. Select option **2** from the Control-M/Server Keystore Management Utility Main Menu.

   The following menu is displayed:

```
+---------------------------------------+
  BLOWFISH Keystore Menu
+---------------------------------------+
1)   Add new key
2)   Apply new key
3)   Restore default key
q)   Quit
Enter option:
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail on the BLOWFISH Keystore utility, see: BLOWFISH Keystore Menu parameters (on page 435).

BLOWFISH Keystore Menu parameters

The following table describes the BLOWFISH Keystore Menu parameters:

Parameter	Description
Add new key	change the encryption key The new key is read from a file and stored. After the key has been stored, all encrypted stored data is decrypted with the old key and then re-encrypted with the new key in a new location, leaving the old data untouched.
Apply new key	replaces the old data with the re-encrypted data This option requires Control-M/Server to be down to replace the old passwords with the new encrypted passwords. New users that were added after the **Add new key** procedure but before the **Apply new key** procedure was carried out, must be reencrypted with new Blowfish key.
Restore default key	re-encrypt all data with default key This option requires Control-M/Server to be down to replace the new passwords with the old passwords.

ctmldnrs

The ctmldnrs utility creates and loads the Manual Conditions file. This file contains prerequisite conditions that are required by jobs in the Active Jobs database but will not be added to the Conditions/Resources table without manual intervention. These conditions fall into two categories:

- Conditions that are never added automatically by scheduled jobs because manual confirmation is always desired.
- Conditions that are normally added automatically by scheduled jobs but the jobs that add them are not scheduled for the day.

Prerequisite conditions in the Manual Conditions file can be made available to the system using the load option of ctmldnrs (see below), using the ctmcontb utility (see: ctmcontb (on page 327)), using the Job prerequisites window, or using the WHY option in the job menu. To run the ctmldnrs, see: Running the ctmldnrs utility (on page 436).

The ctmldnrs utility identifies conditions that should be in the Manual Conditions file by searching for all prerequisite conditions required for submission of jobs on the particular day. The search for prerequisite conditions is performed by checking the In Conditions parameters of the job processing definitions for all jobs in the Active Jobs database. Then, the utility eliminates any "non-manual" conditions that satisfy either of the following criteria:

- The prerequisite condition already exists in the Conditions/Resources table.
- The prerequisite condition is added to the Conditions/Resources table by an Out Conditions or DO COND job processing parameter in a job scheduled to run that day.

Prerequisite conditions that do not meet the above criteria are assumed to be manual conditions and are placed in the Manual Conditions file. To load and create manual conditions file, see: Loading prerequisite conditions from manual conditions file (on page 439) and Creating the manual conditions file (on page 440).

Running the ctmldnrs utility

This procedure describes how to run the ctmldnrs utility, which creates and loads the Manual Conditions file.

➢ To run the ctmldnrs utility:

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter one of the following commands

- `ctmldnrs -LIST <condition-mask> [-INPUT <filename>]`
- `ctmldnrs -LOAD <condition-mask> [-INPUT <filename>]`

-
  ```
  ctmldnrs -CALCCOND [-ADDMODE     {YES | NO}      ]
                     [-OUTPUT      <filename>      ]
                     [-IGNOREIN    <condition-mask>]
                     [-IGNOREOUT   <condition-mask>]
  ```

```
                  [-IGNORECODES <condition-mask>]
                  [-INCLUDE_FUTURE_ODATES       ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information on the ctmldnrs utility, see: ctmldnrs utility options (on page 437) and ctmldnrs utility parameters (on page 438).

ctmldnrs utility options

The following options are available for using this utility:

Parameter	Description
-LIST	List prerequisite conditions from the Manual Conditions file
-LOAD	Load prerequisite conditions from the Manual Conditions file to the Conditions/Resources table
-CALCCOND	Create the Manual Conditions file

By specifying **ctmldnrs -CALCCOND** with the -INCLUDE_FUTURE_ODATES parameter the ctmldnrs utility can process jobs in Wait ODATE (WAIT_ODAT) status and add the appropriate conditions, with the appropriate odate, as if the jobs were in Wait Condition status.

The following special characters are disabled when they occur in prerequisite condition names:

- (open parenthesis
-)   close parenthesis
- | vertical bar
-       space

ctmldnrs utility parameters

The following table describes the ctmldnrs utility parameters:

Parameter	Description
INPUT	Name and full path of a file containing parameters for the utility. In this file, each parameter and its values (if any) are on a separate line with the same syntax they would have on the command line. Using the -input_file parameter enables you to: ▪ prepare and save files of utility parameters that can be reused ▪ specify utility input longer than the number of characters allowed in the command line
ADDMODE	YES – When the new Manual Conditions file is created, conditions from the previous file are retained in the new file. NO – The Manual Conditions file is recreated and all previous conditions are deleted. Default.
OUTPUT	Output file to be created. If this parameter is not specified, the default file is **<***controlmUserDir***>/ctmldnrs.dat**.
<Filename>	Full path name of the output file to be created.
IGNOREIN	All conditions that satisfy the specified condition name are ignored when the file is created.
IGNOREOUT	References to conditions that satisfy a condition name that is specified in OUT COND job processing parameters are ignored by the algorithm that builds the file.
IGNORECODES	References to conditions that satisfy a condition name that is specified in DO COND job processing parameters are ignored by the algorithm that builds the file.
<condition-mask>	Name of the prerequisite condition. The condition name can include the wildcard character * to represent any number of characters (including no characters). In this instance, the condition name must be enclosed in quotation marks (for example, "LVL *"). Specify "*" by itself to include all existing conditions. When using both open and closed square brackets ([and]), the condition name must be enclosed in quotation marks (for example, "RATE[A1]").

Parameter	Description
-INCLUDE_FUTURE_ ODATES	Specifies the action to be taken with jobs that are in Wait ODATE state When this parameter is specified, jobs in Wait ODATE state are included in the utility processing. When this parameter is not specified, jobs in Wait ODATE state are excluded from the processing of the ctmldnrs utility (default). Conditions added by ctmldnrs with a date reference (month and day) that is later than the current Control-M/Server date (ODATE) are deleted by the Control-M/Server New Day processing a day before that date arrives. If this action is not the intended action, set New Day processing to *not* perform old conditions cleanup. You can manually delete the old conditions a few days after they are no longer required. Set the Ignore New Day Conditions parameter to **Y** by using the ctmsys utility. Specify * in the <*Control-MServerHomeDir*>**/ctm_server/data/dbs_ignrcond.dat** file (all the conditions are ignored).

Loading prerequisite conditions from manual conditions file

This procedure describes how to run the load prerequisite conditions from the Manual Conditions file.

➢ To load prerequisite conditions:

1. Do one of the following:

- **a.** Log on to a Control-M/EM account (*UNIX*)
- **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
- **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
ctmldnrs -LOAD <Condition Name> [-INPUT <Filename>]
```

For parameters, see: Manual conditions parameters (on page 441)

Creating the manual conditions file

This procedure describes how to run the create the manual conditions file.

➢ To create the manual conditions file

1. Do one of the following:

   a. Log on to a Control-M/EM account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command

```
ctmldnrs -CALCCOND [-ADDMODE     {YES | NO}      ]
                  [-OUTPUT      <filename>      ]
                  [-IGNOREIN    <condition-mask>]
                  [-IGNOREOUT   <condition-mask>]
                  [-IGNORECODES <condition-mask>]
                  [-INCLUDE_FUTURE_ODATES       ]
```

Manual conditions parameters

The following table describes the conditions from the Manual Conditions file to the Conditions/Resources table:

```
ctmldnrs -LOAD <Condition Name> [-INPUT <Filename>]
```

Parameter	Description
<Condition Name>	All conditions in the input file that satisfy the specified characters are loaded or listed. Specify "*" by itself to load/list all conditions. When using both open and closed square brackets ([and]), the condition name must be enclosed in quotation marks (for example, "RATE[A1]").

<Filename>	Path name of the input Manual Conditions file. If this parameter is not specified, the default input file is:
	UNIX
	`<Control-M Server Home dir>/ctm_server/tmp/ctmldnrs.dat`
	Windows
	`<Control-M Server Home dir>\ctm_server\temp\ctmldnrs.dat`

Manual conditions file example

The following are examples of the manual conditions file;

The following command re-creates the default Manual Conditions file in the user's directory:

```
ctmldnrs -CALCCOND -ADDMODE NO
```

The following command creates a Manual Conditions file **/h/mcond/data/output.dat** that ignores conditions with prefix "a":

```
ctmldnrs -CALCCOND -ADDMODE NO -OUTPUT /h/mcond/data/output.dat  \
-IGNOREIN "a*"  \
```

The following command loads all conditions from the default input Manual Conditions file to the Conditions/Resources table:

```
ctmldnrs -LOAD "*"
```

ctmlog

The ctmlog utility creates a report from entries in the Control-M log or deletes entries in the Control-M log. To run the ctmlog utility, see: Running the ctmlog utility (on page 442).

Running the ctmlog utility

This procedure describes how to run the ctmlog utility, which creates a report from entries in the Control-M log or deletes entries in the Control-M log.

➢ To run the ctmlog utility

1. Do one of the following:
   - **a.** Log on to a Control-M/EM account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter one of the following commands:

   -
     ```
     ctmlog <action> <actionOption>  \
       <fromDate> <fromTime> <toDate> <toTime>  \
       [<output> [<reportWidth>] ]
     ```

   -
     ```
     ctmlog <action> <actionOption> "*"  \
       [<output> [<reportWidth>] ]
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For valid values for *<action>* and *<actionOption>* see ctmlog utility actions (on page 443). All other parameters of this utility, see ctmlog utility parameters (on page 444).

ctmlog utility actions

The following table lists all actions are limited to log entries in the range specified using the time and date parameters:

Action		Action option	
listss	Prints a report for a specific subsystem.	<subsystem>	Subsystem to include in the report. Specify one of the following: ▪ SU Supervisor ▪ TR Tracker ▪ SL Selector ▪ CD Download, Database update ▪ NS Communication with agent computers ▪ LG Agent utilities ▪ UT Utilities ▪ WD Watchdog
list	Prints a report of all entries.	<hours>	When running ctmlog in the list mode and there is a number between 1 to 48 following the list action, then ctmlog treats this number as the number of hours earlier than when the log was specified to be generated. If this number is not within the range between 1 to 48, then the ctmlog utility expects the number to be either in the format yyyymmdd or yymmdd.
listord	Prints a report of entries for a specific Order ID.	<Order ID>	Order ID to include in the report. The Order ID as displayed in the Job Details window of Control-M/EM is a base 36 number. If you want to specify the Order ID here as a base 10 number, precede the number with an asterisk and enclose the result in quotation marks (for example, "*1234").
listjob	Prints a report including all entries for a specific job number.	<Job no.>	Job number to include in the report.
listmsg	Prints a report of messages with a specific message ID.	<msgid>	Message ID to include in the report.

Action		Action option	
delete	Deletes entries in a specified date and time range.	None.	
listjobname	Prints a report including all entries for the specified job name.	<jobName>	The name of the job whose entries should be printed in the report.

ctmlog utility parameters

This table lists the parameters of the ctmlog utility:

Parameter	Description
<From Date> <From Time> <To Date> <To Time>	Starting and ending dates and times for the range of entries to be scanned by the specified action. Date is specified in yyyymmdd format. Time is specified in hhmm format.
"*"	Asterisk enclosed in quotation marks. Scan all entries in the Control-M log (without regard to date or time).
<Output>	Full path name to which the report should be sent (optional). If this parameter is not specified, the output is routed to the default output device. This parameter is not applicable for the delete action.
<Report Width>	Width (in columns) of the report to generate. Specify a number in the range of 80 – 132 (default is 80). This parameter can only be specified if the Output parameter is specified.

ctmlog utility example

The following are examples of ctmlog utility commands:

The following command produces a report of all entries in the Control-M log between 10:00 A.M. March 12, 2008 and 8:00 A.M. March 14, 2008. The report is output to the **rprt.txt** file in 80-column format:

```
ctmlog list 20080312 1000 20080314 0800
~<controlm_owner>/ctm_server/user1/rprt.txt
```

The following command produces a report of all entries in the Control-M log relating to downloads to the Control-M/EM database and to Control-M/Server database updates, without regard to date or time. The report is output to file **gdrprt.txt** in 132-column format:

```
ctmlog listss CD "*" ~<controlm_owner>/ctm_server/user1/gdrprt.txt
132
```

ctmsys

The ctmsys utility is an interactive utility for maintaining:

- Shout Destination tables (for directing Shout messages).
- Control-M system parameters. For more information, see System configuration.

To run the ctmsys utility, see Running the ctmsys utility (on page 445).

Shout Destination tables associate logical output destinations (specified in Shout and Do Shout statements in job processing definitions) with physical destination names. For more information, see the description of the Shout facility in Shout destination management.

You can do the following from the ctmsys utility:

- Creating/Updating a shout destination table (on page 447)
- Shout Destination table parameters (on page 447)
- Creating a new entry in a shout destination table (on page 448)
- Modifying an existing entry in a shout destination table (on page 449)
- Deleting an existing entry in a shout destination table (on page 449)
- Editing an active shout destination table (on page 449)
- Listing existing shout destination tables (on page 450)
- Deleting an existing shout destination table by name (on page 450)
- Accessing the Control-M system parameters from the ctmsys utility (on page 451)

Running the ctmsys utility

This procedure describes how to run the ctmsys utility, which enables you to maintain Shout Destination tables and system parameters.

➢ To run the ctmsys utility:

1. Log on to the server computer as the Control-M for Databases owner (for example, user controlm).
2. Specify the command:

   ```
   ctmsys
   ```

   The following menu is displayed:

   ```
   +-------------------------------------------------+
   |     Control-M SYSTEM MAINTENANCE UTILITY        |
   |                 Main Menu                       |
   +-------------------------------------------------+

   1)   Shout Destination Tables
   2)   System Parameters

   q)   Quit
   ```

```
Enter option:
```

The options in this menu and in all other menus provided by this utility can be selected by typing the option number or command letter and pressing <**Enter>**.

Each option appearing in the Main Menu is described below.

Accessing the shout destination tables menu

This procedure describes how to access the shout destinations table menu, which enables you to create, modify, list, and delete shout destination tables.

➢ To access the shout destination tables menu:

1. Select Option 1 from the Main Menu.

The following menu is displayed:

```
Shout Destination Tables Menu
-----------------------------
Active Shout Destination Table: <tableName>

1)  Create/Modify a Table
2)  Set SMART Folder
3)  List Tables
4)  Delete Table

q)  Quit and return to main menu
```

```
Enter option:
```

The name of the currently-active Shout Destination table is displayed in the <tableName> field on the menu.

2. Modify or view parameters as follows:

- To switch between the two pages of parameters, type n (next page) or p (previous page) as required.
- To modify a parameter, type the number preceding the parameter.
If the parameter has a Y/N value, typing the parameter's option number toggles the value between Y and N and redisplays the page.
- If the parameter requires any other value, you are prompted to type the value. After you supply the value, the page is redisplayed.

3. To exit the utility, do one of the following:

- Type **s** to save your changes and exit to the Main Menu. Modifications are not saved until you perform this action.
- Type **c** to cancel all changes and exit to the Main Menu.

Creating/Updating a shout destination table

This procedure describes how to create/update a shout destination table from the ctmsys utility.

➢ To create or modify a shout destination table:

1. Select Option 1 from the Shout Destination Tables menu.

   A list of available tables, similar to the following, is displayed:

   ```
   Shout Destination Tables
   ------------------------
   SYSTEM
   NIGHT_SHIFT

   Table to create/modify or 'q' to quit [SYSTEM]:
   ```

2. Specify the name of the table to be created or modified (or press **<Enter>** to accept the default).

   If the name you specify is not the name of an existing Shout Destination table, a new table will be created with the specified name.

   A display similar to the following is displayed. For an existing table, the display lists the defined destinations.

   ```
   Shout Destination Table 'SYSTEM'

   --------------------------------

   #  Destination Type  Addr  Logical Name  Physical Name

   -  ----------------  ----  ------------  -------------

   1  E                 S     EM

   2  T                 S     Term_B        $TTB.#B

   q) Quit  e#) Edit entry #  n) New entry  d#) Delete entry #

   Enter option:
   ```

For more information, see Shout Destination table parameters (on page 447).

Shout Destination table parameters

The following table lists the parameters for creating.updating a shout destination table from the ctmsys utility:

Item	Description
#	Entry number in the table.
Destination Type	One-letter code indicating the type of recipient:
	U—Specific user. If the user is not logged onto the data center when the Shout message is sent, the message is placed in the user's mail.
	M – User's mail.

Item	Description
	T – Destination is a specific terminal or file known to the operating system from which the shout was invoked.
	O – System console.
	L – Control-M/Server log.
	E – Alert window of Control-M/EM.
	P – Sends the shout message to a specific program.
Addr	One-letter code indicating whether the destination is on the server (S) or agent (A) computer.
Logical Name	Name used in the Shout or Do Shout parameter of the job processing definition to identify the recipient of the Shout message.
Physical Name	For Destination Type U, name of a user in the data center. For Destination Type M, e-mail address of the user (for example, username@xxx.zzz). For Destination Type T, terminal ID or full path name (max. 96 characters) of a file. If the file exists, the message will be appended to the end of the file. For Destination Type P, the full path name of the program file/script that will perform the Shout operation. For Destination Types O, L, and E, no physical name is specified because each of these is a unique destination.

Creating a new entry in a shout destination table

This procedure describes how to create a new entry in a shout destination table from the ctmsys utility.

➢ To create a new entry in the table:

1. Type n.

   The following prompts appear:

   Dest. Type: (U)ser (M)ail (T)erminal c(O)nsole (L)og (P)rogram Control-M/(E)M:

2. Specify the letter corresponding to the desired destination type.

   The following prompt is displayed:

   Address Type (S)erver or (A)gent:

3. For Destination types U, M, P, T, or O, specify whether the destination is on the server (S) or agent (A). For Destination type E, specify S.

The following prompt is displayed:

Logical Name:

4. Specify the logical name for this destination.

   The following prompt is displayed:

   Physical Name:

5. For Destination types U, M, P, or T, specify the physical name. For Destination types O and E, leave this field blank.

   The new entry is added to the table.

Modifying an existing entry in a shout destination table

This procedure describes how to modify an existing entry in a shout destination table from the ctmsys utility.

➢ To modify an existing entry (physical name only) in the table:

1. Type e<entry_number>. For example, to modify entry number 2, specify e2.

   The following prompt is displayed:

   Dest Type:
   Address Type:
   Physical Name:

   This option cannot be used to modify a logical name.

2. Specify a new physical name for the entry.

   The table is redisplayed with the modified entry.

Deleting an existing entry in a shout destination table

This procedure describes how to delete an existing entry in a shout destination table from the ctmsys utility.

➢ To delete an existing entry in the table:

1. Specify d<entry #>. For example, to delete entry # 2, specify d2.

   The entry is deleted.

2. Specify q to return to the Shout Destination Tables menu.

   The Shout Destination Tables menu is redisplayed.

Editing an active shout destination table

This procedure describes how to edit an active shout destination table from the ctmsys utility.

➢ To edit the active shout destination table:

1. Select Option 2 from the Shout Destination Tables menu.

   A list similar to the following is displayed:

   ```
   Existing Shout Destination Tables
   ```

```
---------------------------------
SYSTEM
NIGHT_SHIFT
Enter name of table to make active or q to quit [SYSTEM]:
```

2. Specify the name of the table to set as the active Shout Destination table.

   The following message is displayed:

```
Table <table name> is now active.
Press ENTER to continue.
```

3. Press <**Enter**> to return to the Shout Destination Tables menu.

The active Shout Destination table is changed immediately, affecting Shout and Do Shout operations performed by Control-M.

To specify the active Shout Destination table using a job, run the ctmshtb utility, described in ctmshtb (on page 393)

Listing existing shout destination tables

This procedure describes how to list existing shout destination tables from the ctmsys utility.

➢ To list existing shout destination tables:

1. Select Option 3 from the Shout Destination Tables menu.

   A list similar to the following is displayed:

```
Shout Destination Tables
------------------------
SYSTEM
NIGHT_SHIFT
Press ENTER to continue
```

2. Press <**Enter**> to return to the Shout Destination Tables menu.

   The Shout Destination Tables menu is redisplayed.

Deleting an existing shout destination table by name

This procedure describes how to delete an existing shout destination table by name.

➢ To delete an existing shout destination table by name:

1. Select Option 4 from the Shout Destination Tables menu.

   A list of existing Shout Destination Tables is displayed.

2. Specify the name of the table to delete.

   The following message is displayed:

Delete completed successfully.
Press ENTER to continue.

3. Press <**Enter**> to return to the Shout Destination Tables menu.

   It is not possible to delete the active Shout Destination table.

Accessing the Control-M system parameters from the ctmsys utility

This procedure describes how to access the Control-M system parameters from the ctmsys utility, which enables you to view or edit Control-M system parameters.

➢ To view or modify Control-M system parameters:

- Choose Option 2 from the Main Menu.

  The first group of parameters (and their current values) is displayed.

  Control-M System Parameters (Page 1)

```
Control-M System Parameters (Page 1/2)
--------------------------------------

Computer System         :sparc
Operating System        :Solaris
Control-M Version       :6.3.0
Database Version/schema :6.3.0
Executable Path :/home3/ctmssl/ctm_server/exe_Solaris
1) Day Time                :+0700
Control-M Date          :20071216
2) Statistics              :Y
3) Maximum Retries         :50
4) Start Day of the Week   :2
 Active Shout Table      :SYSTEM
5) Full Security           :N
n) Next Page
s) Save and Return to Main Menu
c) Cancel
Enter command, or item number you wish to change [n]:
```

  When you specify n, the second page of parameters is displayed.

  Control-M/Server System Parameters (Page 2)

```
Control-M System Parameters (Page 2/2)
--------------------------------------
6) Maximum Days Retained by Control-M Log :2
7) Maximum Days to Retain Output Files    :1
8) Ignore New Day Conditions              :N
9) Secure Sockets Layer                   :DISABLED
p) Previous Page
s) Save and Return to Main Menu
c) Cancel
Enter command, or item number you wish to change [p]:
```

Running the ctmwincfg utility

This procedure describes how to run the ctmwincfg utiltiy.

➢ To run the ctmwincfg utility:

- Type the following from a command prompt.

  ```
  ctmwincfg
  ```

For a description of the parameters in the ctmwincfg utility, see the configuration parameters in System configuration.

This utility can also be accessed as a Java application. For more information, see System configuration.

Running the ctm_remedy_configure utility

This procedure describes how to run the ctm_remedy_configure utility, which enables you to change and test the Do Remedy action settings.

➢ To run the ctm_remedy_configure utility

1. Log on to the Control-M/Server computer as the Control-M/Server owner.
2. To invoke the utility, enter the following command:

   ```
   ctm_remedy_configure
   ```

   The following interactive menu is displayed:

   ```
   1> Basic Parameters

   2> Reset user password

   3> Test configuration

   4> Quit
   ```

3. Do one or more of the following:

   a. Press 1 to change basic settings:

      - Remedy server name
      - Remedy server port
      - Remedy user name

- Login user
- First user name
- Last user name

b. Press 2 to enter a new user password

c. Press 3 to test the changes that were made to the configuration

d. Press 4 to exit the utility

Control-M/Agent utilities

This table lists the Control-M/Agent utilities for administration and configuration.

Utility Type	Description
agkeystore (on page 453)	The agkeystore utility enables you to create, apply, and restore keys to encrypt job owners credentials using a Blowfish alogorithm.
ctmag (on page 458)	The Agent Configuration (ctmag) utility is a Java application used to maintain Control-M/Agent configuration parameters, and to view and modify most of the operating system parameters.
Running the ctmagcfg utility (on page 458)	The ctmagcfg utility is used to interactively configure Control-M/Agents.
Running the ctmunixcfg utility (on page 458)	The interactive ctmunixcfg utility enables you to view and modify most of the configuration parameters in the **OS.dat** file.
set_agent_mode (on page 459)	Use the set_agent_mode utility to enable or disable the non-root mode of Control-M/Agent on UNIX.
Running the ag_change_password utility (on page 460)	Automates password changes of Application Add-on accounts.

agkeystore

The agkeystore utility enables you to create, apply, and restore keys to encrypt job owners credentials using a Blowfish alogorithm. To run the agkeystore utility, see Running the agkeystore utility (on page 454).

Changes to the Blowfish encryption key made using the agkeystore utility must also be made in Control-M/Server using the ctmkeystore_mng utility.

Running the agkeystore utility

This procedure describes how to run the agkeystore utility, which enables you to create, apply, and restore keys to encrypt job owners credentials using a Blowfish alogorithm.

➢ To run the agkeystore utility:

- Do one of the following:
  - To run the utility interactively, type the following command:

    ```
    agkeystore
    ```

    The following menu is displayed:

    ```
    Select agent from the list:
    1) Default
    2) agent-host1
    3) agent-host4
    q) Quit
    Please enter your choice:
    ```

    The options in this menu and in all other menus provided by this utility are selected by typing the option number or command letter and pressing <**Enter**>.

    Assuming that the default agent is chosen, the following menu is displayed:

    ```
    Control-M/Agent Key Store Management Utility
    Agent Name: default
    1) Add new key
    2) Apply new key
    3) Restore default key
    q) Quit
    Enter your choice:
    ```

  - To run the utility silently, type the following command depending on your operating system:
    - **For UNIX:** Specify the following command:

      ```
      agkeystore -key <key type> -action <action_name> -input
      <input_file_full_path> [-replace <Y/N>]
         [-key     key type <BLOWFISH>]
         [-action  <add>  <apply> <restore>]
         [-input   full path of input file containing new blowfish key]
         [-replace replace existing new blowfish key if exists <Y|N>]
      ```

    - **For Windows:** Specify the following command:

      ```
      agkeystore -agent <agent_name> -key <key type> -action <action_name>
      -input <input_file_full_path> [-replace <Y/N>]
      ```

```
[-key     key type <BLOWFISH>]
[-action  <add>  <apply> <restore>]
[-input   full path of input file containing new blowfish key]
[-replace replace existing new blowfish key if exists <Y|N>]
```

agkeystore options

The following table describes the options in the agkeystore utility:

Option	Description
Add new key	Creates a new Blowfish encryption key in the **agkeystore.kdb** file. After selecting the **Add new key** option, you are prompted to enter an input file name to the file that contains the key. Ensure that you specify the full path to this file. If a new Blowfish key was added earlier but was not yet applied by completing the **Apply new key** procedure, you are prompted to either overwrite the earlier Blowfish key with a newer Blowfish key or leave the Blowfish key that was created earlier intact and not create a subsequent newer Blowfish key. When Control-M/Agent is running on Windows computers and configured with **LOGON_AS_USER** logon option, a new PASSWRDS file containing all the job owners and their passwords encrypted with the new key is created. The original PASSWRDS file is not changed. After adding a new key, only use ctmpwd to create and modify Control-M/Agent users and passwords when you have completed the **Apply new key** procedure.
Apply new key	The new key replaces the existing Blowfish encryption key in the **agkeystore.kdb** file used by Control-M/Agent and the PASSWRDS file is replaced with the new PASSWRDS file that is created in the **Add new key** option when Control-M/Agent is running on Windows computers and configured with **LOGON_AS_USER** logon option. Immediately after applying the new key, restart Control-M/Agent for the new Blowfish encryption key to be used.
Restore default key	Replaces the existing Blowfish encryption key entry in the **agkeystore.kdb** file with the default Blowfish encryption key that is used by Control-M. When Control-M/Agent is running on Windows computers and configured with **LOGON_AS_USER** logon option, the PASSWRDS file is saved with the job owners and their passwords encrypted with the default key. Immediately after restoring the default key, restart Control-M/Agent for the new Blowfish encryption key to be used.

agkeystore actions

The following table describes the actions in the agkeystore utility:

Action	Description
-agent	(*Windows only*) Specify the name of the agent whose Blowfish encryption of job owners credentials are being modified.
-key	Specify the type of encryption key. Enter the value **BLOWFISH**
-action	Specify the action that the utility must take on the **agkeystore.kdb** file. The valid values are: add – creates a new Blowfish encryption key in the **agkeystore.kdb** file If a new Blowfish key was added earlier but was not yet applied by completing the **Apply new key** procedure, you can either overwrite the earlier Blowfish key with a newer key or leave the Blowfish key that was created earlier intact and not create a subsequent newer Blowfish key. apply – the updated Blowfish encryption key replaces the existing Blowfish encryption key in the **agkeystore.kdb** file The Control-M/Agent must be restarted for this action to take effect. restore – replaces the existing Blowfish encryption key in the **agkeystore.kdb** file with the Blowfish encryption key that is used by Control-M The Control-M/Agent must be restarted for this action to take effect.
-input	Specify the full path to the file that contains the Blowfish encryption key. The maximum length of the key is 32 characters (256 bits) long.
-replace	Replace a new Blowfish encryption key that was created but that has not yet been applied by completing the **Apply new key** procedure. **-replace** is relevant only when **-action add** is specified and another Blowfish encryption key was added but has not yet been applied. In this case use **-replace** to overwrite the existing Blowfish encryption key entry with the new Blowfish key. The valid values are: ▪ **Y** – replace existing new Blowfish encryption key if it exists ▪ **N** – do not replace the Blowfish encryption key

ctmag

The Agent Configuration (ctmag) utility is a Java application used to maintain Control-M/Agent configuration parameters, and to view and modify most of the operating system parameters. If the user running the utility is not an administrator, changes made to agent configuration parameters will not be saved. Running the ctmag utility interactively is described in Control-M/Agent parameters. To run the ctmag utility, see Running the ctmag utility (on page 458).

Running the ctmag utility

This procedure describes how to run the ctmag utility, which enables you to maintain Control-M/Agent configuration parameters, and to view and modify most of the operating system parameters.

➢ To run the ctmag utility:

- Type one of the following commands:
  - ctmag
  - ctmag <*agentName*> (if more than one Control-M/Agent is installed)

Running the ctmagcfg utility

This procedure describes how to run the ctmagcfg utility, which enables you to interactively configure Control-M/Agents.

This utility can also be accessed as a Java application. For more information, see Control-M/Agent parameters.

➢ To run the ctmgfcg utility:

Type the following command:

```
ctmagcfg
```

Running the ctmunixcfg utility

This procedure describes how to run the ctmunixcfg utility, which enables you to interactively view and modify most of the configuration parameters in the **OS.dat** file.

This utility can also be accessed as a Java application. For more information, see System configuration.

➢ To run the ctmunixcfg utiltiy:

- Type the following command:

```
ctmunixcfg
```

For a description of the parameters in the ctmunixcfg utility, see the descriptions of the configuration parameters in System configuration.

set_agent_mode

Use the set_agent_mode utility to enable or disable the non-root mode of Control-M/Agent on UNIX. The utility changes the permissions of several agent files and directories to allow the agent to work in the mode that is selected. This utility must be run by the root user and only needs to be run once to set a mode. Each time the agent is started (or restarted) it will continue to use the mode that was set until such time that the mode is changed. To run the set_agent_mode utility, see Running the set_agent_mode utility (on page 459).

When Control-M/Agent is started by the agent owner user, only jobs where the agent owner is the job owner can be run. To run jobs with different owners, define the passwords of the owners and then use the set_agent_mode utility to enable non-root mode. Both actions must be performed to allow jobs to be submitted. For more information about how to define the passwords of the owners, see Control-M/Agent security or ctmsetown (on page 587).

Running the set_agent_mode utility

This procedure describes how to run the se_agent_mode utility, which enables you to enable/disable the non-root mode of Control-M/Agent on UNIX.

➢ To run the set_agent_mode utility:

- Do one of the following:
  - Type the following command to run the utility interactively:

    ```
    set_agent_mode
    ```

    For more information on the options, see set_agent_mode interactive options (on page 460).
  - Type the following command to run the utility silently:

    ```
    set_agent_mode -u <agentOwner> -o <Option>
    ```

    The valid values for <*Option*> are 1, 2, or 3, as described below:

    - 1 - enable
    - 2 - disable
    - 3 - prepare for uninstall

set_agent_mode interactive options

The following table lists the set_agent_mode interactive options:

Option	Description
Enable non root mode	enables the agent to run jobs by any job owner and registers it in the **/etc/ctm.conf** file
Disable non root mode	returns the agent to the state it was in after the installation
Prepare the agent for non root uninstall	returns the agent to the state it was in after the installation and removes it from the **/etc/ctm.conf** file

Running the ag_change_password utility

This procedure describes how to run the ag_change_password utility, which enables you to automate password changes of Application Plug-ins accounts.

➢ To run the ag_change_password utility:

- Type the following command:

```
ag_change_password -application_type <application_type>
    [-connection_profile <connection_profile_name>]
    [-host <host_name>]
     -user <user_name>
     -old_password <old_password>
     -new_password <new_password>
    [-agent <agent_instance_name>]
```

For more information on the parameters, see ag_change_password parameters (on page 461).

ag_change_password parameters

The following table describes the parameters in the **ag_change_password** utility:

Parameter	Description
-application_type	The type of Application Add-on installed on this agent. Valid values are: ▪ CLOUD – Control-M for Cloud ▪ DATABASE – Control-M for Databases ▪ ETL_INFA – Control-M for Informatica ▪ FILE_TRANS – Control-M for Advanced File Transfer ▪ PS8 – Control-M for PeopleSoft ▪ SAP (Only from 7.0.00) – Control-M for SAP ▪ SAP_BO – Control-M for SAP Business Objects
-connection_profile	The Application Add-on changes the password in the connection_profile_name connection profile. Optional. If no account is specified, all accounts are checked. The * wildcard is supported. See the note below this table.
-host	Match an account where host field is applicable. This parameter is relevant only when **-application FILE_TRANS** is specified. Optional. The * wildcard is supported. See the note below this table.
-user	User account name. The * wildcard is supported. See the note below this table. When the value for the -application parameter is CLOUD and the application is Amazon Elastic Compute Cloud (Amazon EC2), then use Amazon Access Key instead of the user name.
-old_password	User password.
-new_password	New user password.
-agent	(*Windows only*) Name of the Control-M/Agent. If not specified, the default agent is used. Optional.

Wildcards must be enclosed with single quotations (' ') or with double quotations " " according to the requirements of the UNIX shell in which the **ag_change_password** utility is activated.

Health Check

Control-M Health Check utility scans and collects system information about the environment on which Control-M/EM, BMC Batch Impact Manager, and Control-M/Forecast reside. This information is used by BMC Support in helping to troubleshoot and correct problems. The information gathered is maintained in a compressed hierarchical format that allows for analysis of the collected information.

The Control-M Health Check utilities scan and collect system information about the environment on which the various Control-M components are installed. This information is used by BMC Support in helping to troubleshoot and correct problems. The information gathered is maintained in a compressed hierarchical format that allows for analysis of the collected information.

The following Health Check utilities are available with the following data collectors:

- em_data_collector—for Control-M/EM, BMC Batch Impact Manager, and CONTROL-M/Forecast
- ctms_data_collector—for Control-M/Server
- ctma_data_collector—for Control-M/Agent, Control-M/Application Add-ons, and Control-M Business Process Integration Suite

The Health Check the following parameter types, which allow you to scope the data that you package for Support:

- **Components**—Running Health Check utility with default settings gathers data from all products covered by that utility.
- **Category**—information type groupings, for example, operating system, database, or scheduling. For more information, see Standard Category parameters referred to in Health Check utility parameters (on page 467).
- **Profile**—In general, a Profile is a predefined combination of Components and Categories. For more information, see Profile parameters referred to in Health Check utility parameters (on page 467).

The following options are available for running Health Check:

Running Health Check utility from a command line, without specifying parameters. Such *default* runs are described in Running Health Check utility with default parameters (see below).

- Specifying parameter values in a command line, as described in Running Health Check utility with specifying parameters (on page 463).
- Changing default parameter values in **config.ini**, as described in Running Health Check utility using config.ini parameters (on page 463).
- Running Health Check utility from a batch file, as described in Running Health Check utility from a batch file (on page 464).

The Health Check Utility first reads parameters that are specified in the command line (if any). After which the Health Check Utility reads the parameters that have been defined in the **config.ini** file. Values in the **config.ini** overwrite any parameter specified in the command line. If no category/profile or component values are specified in either the command line or **config.ini** file the Health Check Utility executes a pre-defined default run. The run parameters are determined by the **config.ini** file with the default value of the **days** parameter being 10.

Running Health Check utility with default parameters

This procedure describes how to run the Health Check utility without specifying parameters, which enables you to collect and package the data generally required by BMC Support.

➢ To run a Health Check utility with default parameters:

- Enter the following command:
  ```
  -U <dbo_user> -P <dbo_pwd>
  ```

Running Health Check utility with specifying parameters

This procedure describes how to run the Health Check utility with specifying parameters.

Use this procedure if BMC Support instructs you to run Health Check utility with non-default parameters.

➢ To run Health Check utility specifying parameters:

- From a command line or in a batch file, type the following:

  ```
  <your_data_collector> -U <dbo_user> -P <dbo_pwd>
  ```

  The <your_data_collector> variable is one of the following:

  - em_data_collector
  - ctms_data_collector
  - ctma_data_collector

Running Health Check utility using config.ini parameters

This procedure describes how to run Health Check utility after editing certain parameters values in **config.ini**. If there are parameters for which you always override the defaults, such editing can simplify command line entries and batch files.

➢ To run Health Check utility with reference to config.ini:

1. In **config.ini**, set the parameters that you need.
2. Run a command that launches a Health Check utility according to the following:
   - If you do not need to specify additional parameters, see Running Health Check utility with default parameters (on page 463).
   - If you need to specify additional parameters, see Running Health Check utility with specifying parameters (on page 463).

Default Control-M/EM config.ini file

#categories="OS DB LOG"   #this is only an example. Update to the desired categories list. Make sure there is a blank space between each one and only use upper case

#profiles="DB ENV"   #this is only an example. Update to the desired profiles list. Make sure there is a blank space between each one and only use upper case

#products="EM BIM FOR CTM AG CMSAP CMOAP CMBPI"   #this is only an example. Update to the desired products list. Make sure there is a blank space between each one and only use upper case

batch_mode="n" # lower case "y" or "n"

max_size="50" #max files size

days="2" #time period (days) to collect data

ctmname="%%hostname" #default is local host name

gsrname="%%hostname" #default is local host name

bimname="%%hostname" #default is local host name

hidden="dbo_pwd dba_pwd" #Hidden properties should be defined only through command line!

Theoretically, you could completely customize **config.ini** before any given run or set of runs of Health Check utility. This would enable you to launch a non-default run by entering a command such as the following:

```
em_data_collector -U <dbo_user> -P <dbo_pwd>
```

However, in general, you would probably modify **config.ini** only to specify changes to global parameters such as the following:

- days—the maximum number of days in the past for which the utility gathers the information (default value is 2).
- max_size—determines the maximum size of Health Check utility package files (default value is 50).
- batch_mode—if set to y, enables running a Health Check utility without manual confirmation of the following message:
  "The Health Check utility collects information of the environment on which Control-M products reside . . . ."

Running Health Check utility from a batch file

This procedure describes how to run the Health Check utility from a batch file.

➢ To run Health Check utility from a batch file:

1. In **config.ini**, make sure that batch_mode is set to y. For more information, see Running Health Check utility using config,ini parameters (see above).
2. If a parameter requires *<dbo_pwd>*, create a password file. For a list of such parameters, see Parameter Overview referred to in Health Check utility parameters (on page 467).
3. To create a password file, create a text file with the following string:

   dbo=<dbo_password>

   The dbo_password should be the actual Control-M/EM database owner password.

In order for the Health Check Utility to read the password, the value of -file in *ctms_data_collector -U <dbo_user> -file <password_file_path>* needs to point to the exact password file location.

4. In the batch file, type the command or commands that you need. For more information, see To create a password file, create a text file with the following string above.
5. Schedule or run the batch file.

For Control-M/EM, BMC Batch Impact Manager, and Control-M/Forecast, the following command collects data by the Environment diagnostic data collection Profile (**ENV**) for the last five days:

```
em_data_collector -U <dbo_user> -file <PW_file_path> -days=5 -F ENV
```

Check log file

Running a Health Check utility creates a log file that specifies what processes ran and what information was written to the package file. The contents of the log file indicate what proprietary information is included in the package file.

Health Check utility writes log files to the following directories:

- CONTROL-M/ EM—<EM_HOME>/log
- CONTROL-M/ Server—<SERVER_HOME>/ctm_server/health_check/log
- CONTROL-M/ Agent—<AGENT_HOME>/proclog

The log file name is constructed as follows:

<your_data_collector>_<date>_<time>_<osType>_<computerName>_
display.log

where <your_data_collector> is the relevant of the following:

- em_data_collector
- ctms_data_collector
- ctma_data_collector

The Health Check utility package files are saved to the following directories:

- CONTROL-M/ EM—<EM_HOME>/hcu_package
- CONTROL-M/ Server—<SERVER_HOME>/hcu_package
- CONTROL-M/ Agent—<AGENT_HOME>/hcu_package

The package files names are constructed as follows:

- CONTROL-M/ EM—em_data_<TIME_STAMP>_<OS_TYPE>_<HOSTNAME>.zip
- CONTROL-M/ Server—ctms_data_<TIME_STAMP>_<OS_TYPE>_<HOSTNAME>.zip
- CONTROL-M/ Agent—ctma_data_<TIME_STAMP>_<OS_TYPE>_<HOSTNAME>.zip

On UNIX, substitute **tar.z** instead of **zip**.

Examples, tips, and best practices

The explanations for the sample commands are based on the assumption that the shipped version of **config.ini** has not been edited. For more information, see Running Health Check utility using config.ini parameters above.

The following are example commands for running the Health Check utility.

- For Control-M/EM, BMC Batch Impact Manager, and Control-M/Forecast, the following command collects data by all Categories:

  em_data_collector -U *<dbo_user>* -P *<dbo_pwd>*

- For Control-M/EM, BMC Batch Impact Manager, and Control-M/Forecast, the following command collects data by all Profiles for the last seven days:

  em_data_collector -U *<dbo_user>* -P *<dbo_pwd>* -days=7 -F ALL

- The following command collects information by the OS and DB Categories for Control-M/EM, and runs a trace to troubleshoot the utility's execution:

  em_data_collector -U *<dbo_user>* -P *<dbo_pwd>* -trace -C OS DB -D EM

- For troubleshooting the installation of Control-M/EM, BMC Batch Impact Manager, and Control-M/Forecast, the following command collects data by the INST Category:

  em_data_collector -U *<dbo_user>* -P *<dbo_pwd>* -C INST –D EM -D BIM -D FOR

Health Check utility parameters

The following table lists the Health Check utility parameters:

Parameter	Description
passwordInfo	For Categories and Profiles that access a Control-M database, you need to specify *<dbo_user>* and *<dbo_pwd>*. This means the following: Running Health Check utility with default parameters: Control-M/EM—requires dbo parameters Running an Health Check utility with non-default parameters: Categories that do *not* require dbo parameters: OS, CNF, LOG Categories that require dbo parameters: DB, DBT, SCH, TBL, COM All Profiles require dbo parameters.
Components	(optional) Products and applications for which the utility can gather data.
Categories	(optional) Information type groupings, for example, operating system, database, and scheduling. For more information, see Standard Category parameters below.
Profiles	(optional) In general, a Profile is a predefined combination of Components and Categories. If you specify one or more Profiles, do *not* specify Components or Categories. For more information, see Profile parameters below.

Parameter	Description
runConfiguration	You can optionally specify one or more of the following: ▪ -simulate Simulates execution of a Health Check utility. This allows the user to verify that specified parameter values are appropriate. ▪ -days (default: **2**) The maximum number of days in the past for which the utility gathers the information. This option only affects log files relevant to BMC Software components. ▪ -max_size (default: **50**) In megabytes, the maximum size of the compressed file packed by a Health Check utility. ▪ -trace Run a trace on the Health Check utility run. This generates a log file named **ctm_data_collector_<***YYYYMMDD***>_<***HHMMSS***>_<***PLATFORM_TYPE***>_<***HOSTNAME***>.log** ▪ -verbose Outputs the utility processes to your display. You must also specify -trace. ▪ -help or -h Displays the utility's usage. ▪ -file *<PW_file_path>* (batch mode) The path to a file that you created that contains the Control-M/EM database owner (DBO) password. For more information, see Running Health Check utility from a batch file referred to in Health Check (on page 462).

Control-M/EM Health Check utility component parameters

The following table lists the valid component parameters:

Utility	Component Parameter
CONTROL-M/EM Health Check utility	▪ **EM**—CONTROL-M/Enterprise Manager ▪ **BIM**—BMC Batch Impact Manager ▪ **FOR**—CONTROL-M/Forecast
CONTROL-M/ Server Health Check utility	**CTM**—CONTROL-M/Server
CONTROL-M/ Agent Health Check utility	▪ **AG**—CONTROL-M/Agent ▪ **CMAFT**—CONTROL-M for AFT ▪ **CMSAP**—CONTROL-M for SAP ▪ **CMPSFT**—CONTROL-M for PeopleSoft ▪ **CMOAP**—CONTROL-M for Oracle Applications ▪ **CMBPI**—CONTROL-M/Business Process Integration Suite

Control-M/EM Health Check utility installation category parameters

The following table lists Installation Category parameters. These parameters group information from various parts of your system.

Category	Definition
INST (UNIX)	For all components handled, collects the content of the following: ▪ **BMCINSTALL/installed/** ▪ **BMCINSTALL/log/** ▪ **BMCINSTALL/uninstall/** Collects all files with user ownership from the **/tmp** directory. Lists all directories and folders under the component installation path.
INST (Windows)	Collects the content of the following: ▪ **%temp%\PG*.txt** ▪ **%ALLUSERSPROFILE%\Application Data\PG*.txt** ▪ *<EM_HOME>***\bin\DBUtils\DBUData\log\** ▪ **%temp%*_log.txt** ▪ **%temp%*_win.txt** Lists all directories and folders under the component installation path.

Running Health Check utility with **INST** will gather significant information only if the error situation occurred *after* the file copying phase of the installation.

If you specify the **INST** Category parameter, you must also specify one or more component parameters.

If you specify the **INST** Category parameter, do *not* specify any of the following:

Categories other than **INST**

Profiles

Control-M/EM Health Check utility standard category parameters

The following table lists the standard category parameters:

Category	Definition
OS	Operating-system related physical resource. This information includes the following: ▪ operating system version ▪ software and hotfixes installed ▪ environment variables and files ▪ system resources limits ▪ kernel parameters ▪ swap space ▪ system and application logs ▪ scheduled tasks ▪ ini files (Windows) ▪ resource consumption ▪ list of running processes ▪ network settings
DB	Database environment. This information includes the following: ▪ database files—***.ini** and configuration files and database log files ▪ database information collected from SQL queries and configuration parameters For Control-M/EM installed on a UNIX operating system, the Health Check utility gathers **DB** Category data for Sybase, Oracle, or PostgreSQL databases. For Control-M/EM installed on a Windows operating system, the Health Check utility gathers **DB** Category data for MSSQL or PostgreSQL databases.

Category	Definition
DBT	Collects information from the following Control-M/EM database folders: ▪ global_cond ▪ gcs_gtw_recov ▪ gcs_msgs ▪ gcs_dsts ▪ gcs_admin ▪ comm, confreg ▪ logreg ▪ commreg ▪ params ▪ name_value ▪ exception_alerts ▪ download ▪ stat_ex_period – BIM only ▪ stat_ex – BIM only ▪ dictionary – BIM only ▪ time_index – Forecast only ▪ date_index – Forecast only ▪ sim_exception_conditions – Forecast only ▪ sim_sysstate_ud – Forecast only ▪ sim_sysstate_udtbl – Forecast only
SCH	Component scheduling entities measurement. This information includes the following: ▪ number of daily jobs per Control-M ▪ number of daily executions per Control-M

Category	Definition
TBL	Component infrastructure entities measurement. This information includes sizes of the following tables: ■ Alarm ■ conditions and resources tables ■ definition and active job folders ■ global_cond ■ gcs_gtw_recov ■ gcs_msgs ■ gcs_dsts ■ gcs_admin ■ comm ■ confreg ■ logreg ■ commreg ■ name_value ■ audit_activity ■ exception_alerts ■ password_history ■ audit_operations ■ audit_atributes ■ bim_log – BIM only ■ stat_ex_period – BIM only ■ stat_ex – BIM only ■ bim_alert – BIM only ■ dictionary – BIM only ■ dbversion ■ ctmdbcheck

Category	Definition
CNF	Component configuration files. This information includes the following configuration files: ▪ component version (collects the information from **installed-version.txt** file located in the <*emHome*> directory. ▪ the list of files under the <*emHome*> directory. ▪ Installation_Paramaeters.txt ▪ ***.ini** ▪ ***.rsc** ▪ **TAO configuration files**
LOG	Component general diagnostic data collection. This information includes the following: ▪ All component logs for Control-M/EM, BMC Batch Impact Manager, and Control-M/Forecast ▪ A list of the core files created
COM	Component functionality, communication, and connectivity. This information includes information about the communication status and corba availability status.

If you do not specify any Category parameter, the Health Check utility collects data for all Categories.

In general, if you specify a Category, you must also specify one or more components. However, for the following Categories, do *not* specify a component:

- **OS**
- **DB**

You can also specify multiple Categories. In such a case, for each Category the utility gathers data by each relevant component specified.

Control-M for Oracle standard category parameters

The following table lists the Control-M for Oracle standard category parameters

Category	Definition
CNF	Component configuration data. This information includes the following: UNIX: ■ in **ctm/data**: **OAP*.dat** ■ **tnsnames.ora** ■ version of oci library file ■ *<AGENT_HOME>***/installed-versions.txt** Windows: ■ in *<AGENT_HOME>***\DATA**: **OAP*.dat** ■ **tnsnames.ora** ■ version of **oci.dll** ■ *<AGENT_HOME>***\installed-versions.txt** ■ All the registry information under the key: **My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\BMC Software\Control-M/Agent\OAP**
LOG	Component general diagnostic data collection. This information includes the content of the following directories: UNIX: ■ **cm/OAP/data** Windows: ■ **cm\OAP\data**

Control-M/EM Health Check utility profile parameters

The following table lists the profile parameters, which is a predefined combination of Components and Categories. There is also an ENV Profile, which collects data.

If you specify one or more Profiles, do *not* specify Components or Categories.

Profile	Definition
ENV	Collects environment data. This includes information from the following Control-M/EM Categories: ▪ OS ▪ DB
EM	Collects Control-M/EM data. This includes information from the following Control-M/EM Categories: ▪ COM ▪ LOG ▪ DBT ▪ SCH ▪ TBL ▪ CNF
BIM	Collects BMC Batch Impact Manager data. This includes information from the following Control-M/EM Categories: ▪ COM ▪ LOG ▪ DBT ▪ SCH ▪ TBL ▪ CNF

Profile	Definition
FOR	Collects Control-M/Forecast data. This includes information from the following Control-M/EM Categories: ▪ COM ▪ LOG ▪ DBT ▪ SCH ▪ TBL ▪ CNF
ALL	Collects data through all Control-M/EM Health Check utility Profiles, as follows: ▪ ENV ▪ EM ▪ BIM ▪ FOR

purge_runinfo

The purge_runinfo utility can be executed periodically to clean out run information retained by Control-M/Forecast for the purposes of performing its calculations.

The utility is located in one of the following directories, depending on your operating system:

- **EMHome\bin** (on Windows)
- **EMHome/scripts** (on UNIX)

The log files of the purge_runinfo utility are located in the installation log directory both in UNIX and in Windows. The log file that shows the run flow of the utility is called **purge_runinfo_run.log**.

You can run purge_runinfo to clean out Control-M/Forecast run information.

Running the purge_runinfo utility

This procedure describes how to run the purge_info utility, which enables you to clean out run information retained by Control-M/Forecast for the purposes of performing its calculations.

➢ To run the purge_runinfo utility:

- Do one of the following
  - **To run the utility interactively, type the following commands depending on your operating system:**

- **On Windows**

  ```
  <emHome>\bin\purge_runinfo.bat
  ```

- **On UNIX**

  ```
  <emHome>/scripts/purge_runinfo shell script
  ```

- Answer the prompts when they are displayed. You are prompted for the following information:
  - The number of days to retain the information about the run of the jobs
  - The Control-M/EM database owner (DBO) password user name and password

The utility has finished when the message Ended successfully is displayed.

- **To run the utility silently, type the following command:**

```
purge_runinfo –U <emUser> -P <emPass> -keep_days <numDaysRetain>
```

On UNIX, you can hide the password using the file method (in this example, X is the name of the file that contains the password):

```
cat X | purge_runinfo –U <emUser> -P <emPass> -keep_days
<numDaysRetain>
```

`<numDaysRetain>` is the number of days to retain the statistics. If `<numDaysRetain>` is 2, all statistics prior to two days before the current date will be deleted. For example:

```
purge_runinfo –U myuser -P mypassword -keep_days 2
```

The DeleteChunkSize system parameter determines the number of records deleted in one transaction by the purge_runinfo utility when removing run information from the RUNINFO_HISTORY table in the Control-M/EM database. If the DeleteChunkSize value is smaller than the parameter value, no data is deleted.

Database maintenance

The database maintenance utilities can be used to maintain the various Control-M databases and to import and export information to and from additional databases.

Many of the tasks performed by the database maintenance utilities can also be performed using the Control-M Configuration Manager or the root menu. However, by including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utilities used to maintain the various Control-M databases

Utility	Description
build_db (on page 482)	Enables you to define a Control-M/EM Sybase database, or configure a Sybase database client if it became corrupted or if you want to move the database to a different server.
ctm_backup_bcp (on page 532)	Exports data from a Control-M/Server database to the **~<controlmOwner>/ctm_server/backup_db** directory.
ctm_restore_bcp (on page 533)	Imports the Control-M/Server database from the bcp_backup directory.
ctmcheckmirror (on page 534)	Checks the mirroring of the Control-M/Server database and displays the status.
ctmdbbck (on page 535)	Creates back-ups of the Control-M/Server database.
ctmdbcheck (on page 539)	Displays information about the memory capacity and the status of the Control-M/Server database.
ctmdbmused (on page 543)	Returns the size of a Control-M/Server Master database (*Sybase only*) and the percent used.
ctmdbopt (on page 544)	Calculates database statistics.
ctmdbrst (on page 544)	Restores the Control-M/Server database in certain cases.

Utility	**Description**
ctmdbspace (on page 546)	Checks the data and log usage in the Control-M/Server database and displays the usage.
ctmdbtrans (on page 547)	Lists the active transactions in the Control-M/Server database.
ctmdbused (on page 548)	Displays the size (in MB), amount, and percentage of current space usage in the Control-M/Server database data and log.
ctmreindex (on page 550)	Accesses the Control-M/Server database, reads the data dictionary, reads the index definitions, and then reorganizes indexes. (*UNIX only*)
db_check (on page 486)	Provides information about the database. (*Sybase only*)
db_check_space (on page 488)	Provides information for Sybase Adaptive and Oracle database servers. (*Sybase and Oracle only*)
dbversion (on page 550)	Displays a general description of the Control-M/Server database in use, including the current version number.
em_database_menu (on page 490)	Provides an interactive menu to facilitate day-to-day maintenance and diagnostics of Control-M/EM running with PostgreSQL database.
em_rebuild_database (on page 504)	Rebuilds the Control-M/EM database in the specified PostgreSQL server database.
em_SQL (on page 507)	Enables you to connect to the database from the command line.
loader (on page 507)	Used to load default data to the Control-M/EM database.
sweep (on page 509)	Deletes jobs that are no longer active from the Control-M/EM and Control-M/Server databases.
Interactive database utilities (on page 551)	Utilities that can be invoked interactively to facilitate day-to-day maintenance and diagnostics of Control-M/EM running with a PostgreSQL database.
util (on page 517)	Multi-purpose utility that can perform various functions from the command line.

Control-M/EM utilities

This table lists the Control-M/EM utilities for database maintenance.

Utility Type	Description
build_db (on page 482)	The build_db (build database) utility enables you to define a Control-M/EM Sybase database, or configure a Sybase database client if they become corrupted or if you want to move the database to a different server.
db_check (on page 486)	The db_check utility provides the following information about Control-M/EM databases: ▪ Size of the database ▪ Availability of space in the database ▪ Verification of database integrity. ▪ Automatic database and transaction log monitoring
db_check_space (on page 488)	The db_check_space utility provides the following information for Control-M/EM databases that use Sybase Adaptive and Oracle database servers.
em_database_menu (on page 490)	The em_database_menu utility can be invoked interactively to facilitate day-to-day maintenance and diagnostics of Control-M for Databases running with PostgreSQL, Oracle and MSSQL databases.
em_rebuild_database (on page 504)	The em_rebuild_database utility can be invoked to rebuild the Control-M/EM database in the specified PostgreSQL server database.
em_SQL (on page 507)	The em_SQL utility enables you to connect to the database from the command line.
loader (on page 507)	The loader utility is used to load default data to the Control-M/EM database.
sweep (on page 509)	The sweep utility deletes jobs that are no longer active from the Control-M/EM and Control-M/Server databases.

Utility Type	Description
util (on page 517)	util is a multi-purpose utility that can perform the following functions from the command line: ▪ Export data from the Control-M/EM database ▪ Import data to the Control-M/EM database ▪ Delete the Control-M/EM database ▪ Clear the Control-M/EM database ▪ Build the Control-M/EM database ▪ Export a specified definition folder ▪ Import a specified definition folder ▪ Import Control-M/Forecast data ▪ Export Control-M/Forecast data ▪ Remove Control-M/Forecast data (to save disk space)

build_db

The build_db (build database) utility enables you to define a Control-M/EM Sybase database, or configure a Sybase database client if they become corrupted or if you want to move the database to a different server. To run the utility, see Creating a new database on an existing Sybase server (on page 482).

The following build database modes are available:

- **Existing**: Defines a Control-M/EM database on an existing Sybase database server.
- **Client**: Configures a Sybase client so it can communicate with a remote Sybase server. This mode can be used only after the Control-M/EM database has been defined on the Sybase database server.

Creating a new database on an existing Sybase server

The following procedure describes how to create a new Control-M Sybase database on an existing Sybase database server. The existing Sybase database server must be running during the creation of the Control-M/EM database.

If you are recreating an existing Sybase database that was corrupted, you must first perform a cleanup of the corrupted database components. For more information, see Cleaning up existing database (on page 483).

➢ To Create a new database on an existing Sybase database server:

1. Obtain the System Administrator password.
2. Obtain a database name, a System Administrator logon name, and a System Administrator password for the new Control-M/EM database. Use the sp_helpdb command on the remote server to verify that the database name is unique for the Sybase database server.

When creating more than one database on the same Sybase database server, the database name and the database owner name must be unique for each database.

3. Verify that the device file you want to define does not exist by running the following command on the remote server:

```
ls filename .
```

4. Use the following command on the remote server to check the values of the parameters in the following table: sp_configure '<*parameterName*>'. If you change any of these parameters, the change is not implemented until you restart the database.

   A build_db log is created in the home directory of the Control-M for Databases account:

   ```
   <homeDir> BMCINSTALL/log/DRXX.<versionNumber>_user.log
   ```

   `<homeDir>` is the path of the Control-M for Databases user and `<versionNumber>` is the Control-M for Databases version number.

Cleaning up existing database

This procedure describes how to clean up an existing Sybase database**.**

➢ To clean up the database:

1. Log on to the Sybase database server as the sa user.
2. Use the sp_helpdb command to determine if the database exists.
3. If the database exists, use the following command to drop (remove) the database and database owner (DBO):

   drop database <*databaseName*>

   sp_droplogin <*dboName*>

4. Use the following command to check which devices were dropped from the server:

   sp_helpdevice

   go

5. If data or log devices associated with the Control-M database are still listed, drop them using the following command:

   sp_dropdevice <*deviceName*>

   go

6. Delete the files, if any, that are associated with the dropped elements.

   Creating a Sybase database on an existing database server

7. Prepare the information you will need for running the build_db utility. Parameters for this utility are listed in the table below.
8. From the command line of the account on which the existing Sybase database server resides, enter the command: **build_db/build_db.sh**
9. When asked to select the "Control-M component for which you want to install the Sybase component", select **Control-M/Enterprise Manager**.

10. Select the Existing build database mode to create an empty Control-M/EM database on an existing Sybase database server. The name and location of the existing Sybase database server and the system administrator's name and password must be specified.
11. Follow the instructions on the screen, filling in the prompts as necessary. Upon completion, the following message is displayed:

    Installation of Control-M Sybase database component completed successfully.

    For more detail about Control-M Sybase database server parameters, see: Sybase database server parameters (on page 484).

Sybase database server parameters

The following table lists the Sybase database server minimum values parameters:

Parameter	Small < 80 MB	Medium 80-200 MB	Large > 200 MB
Number of locks	30,000	30,000	40,000
Number of User Connections Total number of connections to the Sybase database server required for all applications including Control-M for Databases	60	100	200
Max Memory	32, 768	40,000	100,000

Configuring a sybase client

This procedure describes how to configure a Sybase client

➢ To configure a Sybase client:

1. Prepare the information you will need before running the build_db utility. Parameters for this utility are listed in the table below.
2. From the command line of the account on which you want to create the Sybase database client, enter the command: build_db/build_db.sh
3. When asked to select the "Control-M component for which you want to install the Sybase component", select **Control-M/Enterprise Manager**.
4. Select the Client build database mode. This mode configures a Control-M Sybase database client on the local computer after verifying the existence of a Control-M/EM database on an existing database server.
5. Follow the instructions on the screen, filling in the prompts as necessary. Upon completion, the following message is displayed:

```
Installation of Control-M Sybase database component completed successfully.
```

For more detail on Configuring a Sybase client, see: build_db parameters (on page 485).

build_db parameters

The following table lists the build_db parameters. The most recently used value for each parameter is its default. Determine the required values before running this utility.

Parameter	Description
Sybase Server Host Name	Host name of the computer on which the Sybase Database server is created.
Database Server Administrator Password	Password of the Control-M Sybase database server administrator
Database Name	Control-M/EM database name. Maximum length: 30 characters. The name must begin with an alphabetic character. No spaces are allowed.
Database Owner Login	Logon name of the Control-M/EM database owner.
Database Owner Password	Logon password of the Control-M/EM database owner.
Sybase Server Query Port Number	The port that the computer on which the Control-M Sybase database server is created uses for external communication.
Sybase Server Backup Port Number	The port that the Control-M Sybase database backup server component uses for external communication
Master Device Full Path File Name	Full path name of the Control-M Sybase database server.
Sybsystemprocs Device Full Path File Name	Full path name of the Control-M Sybase database server system processes database.
Temporary Device Full Path File Name	Full path name of the Control-M Sybase database server temporary database.
Data Device Full Path File Name	Full path name, including the logical device name, of the Control-M/EM database. You must specify a new filename in an existing directory path on the computer where the Control-M Sybase database server is created.
Log Device Full Path File Name	Full path name, including the logical device name, of the database log. You must specify a new filename in an existing directory path on the computer where the Control-M Sybase database server is created.
Database Size	Size of the Control-M/EM database in megabytes.
Data Device Size	Space in MB for the data portion of the Control-M/EM database.
Log Device Size	Space in MB for the transaction log. BMC recommends that 30% of the space be allocated for the Control-M/EM database.

Parameter	Description
Temporary Device File Size	Size of the temporary database in megabytes.

db_check

The db_check utility provides the following information about Control-M/EM databases:

- Size of the database
- Availability of space in the database
- Verification of database integrity
- Automatic database and transaction log monitoring

The db_check utility can only be invoked in Control-M/Enterprise Manager.

To run the db_check utility, see Running the db_check utility (on page 486).

When db_check is invoked, information similar to the following is displayed:

Running the db_check utility

This procedure describes how to run the db_check utility, which provides information about Control-M/EM databases.

➢ To run the db_check utility:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
db_check [-d <dbThreshold%>] [-l <logThreshold%>] [-p <password>] [-n] [-h]
```

   The user name is derived from the $EM_USER environment variable.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on db_check utility, see: db check utility parameters (on page 487) and db_check utility example (on page 487).

db check utility parameters

The following table describes the db_check utility parameters:

Parameter	Description
-d <dbThreshold%>	Maximum percentage of database use. When this percentage is exceeded, a message is displayed alerting you to extend the database. The -d must be lowercase.
-l <logThreshold%>	Maximum percentage of transaction log use. When this percentage is exceeded, a message is displayed alerting you to extend the transaction log. The -l must be lowercase.
-p <password>	Password for the Control-M/EM administrator. If not specified, you are prompted to supply this information when the utility runs. The -p must be lowercase.
-n	When -n is specified, db_check is executed without verifying the total database integrity. The -n must be specified in lowercase.
-h	When -h is specified, db_check displays the amount of database space that is in use. The -h must be specified in lowercase.

db_check utility example

The following is an example of the db_check utility:

```
db total = 29000.0 KB (data= 23500.00, log= 5500.00)

data used = 1928 KB (8%).

log used = 0 KB (0%).

Checking database...

Database is OK.
```

The db_check utility is not relevant for Windows and works only with databases on a Sybase Adaptive server. This utility is not available for Oracle server.

db_check_space

The db_check_space utility provides the following information for Control-M/EM databases that use Sybase Adaptive and Oracle database servers.

- Total size and availability of space in the **/tmp** directory
- Total size and availability of space in the database
- Total size and availability of space in the log (*Sybase only*)
- Percentage of total space in the database that is currently available

The db_check_space utility cannot be used for Oracle database client installations and is not relevant for Windows. The db_check_space utility can only be invoked in Control-M/Enterprise Manager.

To run the db_check space utility, see: Running the db_check_space_utility (on page 488).

Running the db_check_space_utility

This procedure describes how to run the db_check_space utility, which provides information for Control-M/EM databases that use Sybase Adaptive and Oracle database servers.

➢ To run the db_check_space utility:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Do one of the following:

   - When prompted by the utility interactively:
     - For Oracle, specify the Oracle administrator as the user name.
     - For Sybase, specify the database owner, any database user, or the Sybase Adaptive server administrator as the user name.
   - Enter the following command to specify the appropriate Control-M/EM UNIX user account and password using the -U and -P parameters in the command line:

```
db_check_space [ -U <userName> [ -P <password> ] ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the db_check_space utility, see: db_check_space utility parameters (on page 489) and db_check_space utility example (on page 489).

db_check_space utility parameters

The following table describes the db_check_space utility parameters:

Parameter	Description
-U <userName>	User name. If not specified in the command line, you are prompted to provide this information when the utility runs. ▪ The -U must be specified in uppercase. ▪ If you intend to specify the -P parameter when running the utility, you must also specify the -U parameter.
-P <password>	Password. If not specified in the command line, you are prompted to provide this information when the utility runs. ▪ The -P must be specified in uppercase. ▪ If you intend to specify the -P parameter when running the utility, you must also specify the -U parameter.

This utility is also available from the root menu by selecting the 2 - Troubleshooting Menu option and then the 1 - Database Troubleshooting option. For more information, see Control-M diagnostics.

db_check_space utility example

The following are examples of db_check_space utility output:

Sybase Adaptive server

```
        +------------------------------------------+
        + Space Information ( /tmp of the machine) +
        +------------------------------------------+
         Total Size:  524288 KB.  Space Available:  401548 KB  23% Free
        +-------------------------------------+
        |  SYBASE Size Information:           |
        +-------------------------------------+
        Type            Total Size    Free Space      % Free Space
        ----------------------------------------------------------
        Data :          80 MB         72944 KB        89.043 %
        Log :           23 MB         23458 KB        99  %
```

Oracle server

```
        +------------------------------------------+
+ Space Information ( /tmp of the machine) +
+-------------------------------------------+
 Total Size:  2048000 KB.  Space Available:  366430 KB  17 % Free
+-------------------------------------+
|  ORACLE Size Information:           |
+-------------------------------------+
 db total = 276480.0 KB
data used = 25296 KB (9%).\
```

em_database_menu

The em_database_menu utility can be invoked interactively to facilitate day-to-day maintenance and diagnostics of Control-M for Databases running with PostgreSQL, Oracle and MSSQL databases.

On Windows, the utility is called **em_database_menu.bat.**

The Database Utilities Menu is only relevant for PostgreSQL, Oracle and MSSQL database functionality.

The Database Utilities menu and the resulting sub-menus are described in the following:

- Accessing the Database Utilities Menu using PostgreSQL (on page 492)
- Accessing the Database Utilities Menu using Oracle (on page 499)
- Accessing the Database Utilities Menu using MSSQL (on page 501)

The sub-menus and certain options from the em_database_menu utility can be invoked from the command line. For database menu options see em_database_menu options (on page 491).

em_database_menu options

The following table describes the utilities with the related options from the em_database_menu:

Utility	em_database_menu sub-menu or option
Database Utilities Menu	
dbu_menu	Database Utilities Menu
Management Menu (only available for PostgreSQL databases)	
DBUStart	Start database
DBUStop	Stop database
DBUArchive	Set database archive mode
Mainentance Menu	
DBUMaintain	Maintain Database
DBUColdBackup	Cold Database Backup
DBUHotBackup	Hot Database Backup
DBUHotRestore	Hot Database Restore
DBUColdRestore	Cold Database Restore
DBUBuild	Build Database option in the Database Creation Menu
Report Menu	
DBUStatus	Database Status
DBUShow	Database Parameters
DBUStorage	Database Storage Report
DBUVersion	Database Version
DBUTransactions	List All Active Transactions

Accessing the Database Utilities Menu using PostgreSQL

This procedure describes how to access the Database Utilites menu using PostgreSQL.

➢ To access the Database Utilities Menu:

- Type the following command at the command prompt:

  ```
  em_database_menu
  ```

  The Database Utilities Menu is displayed:

  ```
  Select one of the following options:
  1 - Management
  2 - Maintenance
  3 - Report
  q - quit
  Enter option number --->   [q]:
  ```

  The options available in the Database Utilities Menu enable you to access the following sub-menus:

  - Management Menu options (on page 493)
  - Management Menu options (on page 493)
  - Report Menu options (on page 498)

  Within these sub-menus are a variety of functions used to maintain the Control-M database.

Management Menu options

The following table lists the options in the Management Menu. The Management Menu is not available when using Oracle or MSSQL databases.

Option	Description
Start Database	**Only enabled when Control-M/EM is running with a dedicated database server.** Starts the database server, and services involved. This option is only enabled on Control-M/EM running with a dedicated PostgreSQL database server. If this option is invoked on Control-M/EM running with an existing database, an error message is displayed.
Stop Database	**Only enabled when Control-M/EM is running with a dedicated database server.** Stops the database server, and services involved. This utility is only enabled on Control-M/EM running with a dedicated PostgreSQL database server. **Force:** A parameter that enables you to abort the database server processes and listener. **Valid values**: ▪ Y ▪ N (default). If this option is invoked on Control-M/EM running with an existing database, an error message is displayed.
Set Database Archive Mode	**Only enabled when Control-M/EM is running with a dedicated database server.** Changes the archive mode of the database server. **Parameters available:** **Mode** ▪ On ▪ Off (default) **Archive Directory (only available when Mode is ON)** Full directory path to which the database server logs are archived. This directory should be empty. When Archive mode is **On**, the **Hot Database Backup** option is enabled.

Option	Description
Build Database	Creates a Control-M/EM schema on the database server. This option is only enabled on Control-M/EM running with either an existing or a dedicated database server.

Maintenance Menu options

The following table lists the Maintenance Menu options:

Option	Description
Maintain Database	Activates statistic procedures in the database server.
Hot Database Backup	Only enabled when: ▪ **Database Archive Mode is set to On** ▪ **Control-M/EM is running with a dedicated database server.** Online backup of the database server file system, and the Control-M/EM databases. This can be used to restore the database up to and including the last completed transaction. **Parameters available:** **Backup Directory** Full path to the directory into which the server file system and Control-M/EM databases should be backed up. This directory should be empty. **Administrator password** Password of the database server administrator. This option is only available when using a PostgreSQL database.
Cold Database Backup	Exports the Control-M/EM database schema to the specified file. The following Control-M/EM components must be shut down prior to activating this backup: ▪ Gateway ▪ GUI server ▪ BMC Batch Impact Manager ▪ Control-M/Forecast ▪ Global Conditions Server ▪ Control-M Configuration Management Server ▪ Control-M Configuration Agent **Parameters available:** **Backup File** Full path to the file into which the database should be backed up. **Administrator password** Password of the database server administrator.

Option	Description

Hot Database Restore	Only enabled when Control-M/EM is running with a dedicated database server.
	Online restore of the PostgreSQL database server file system and the Control-M/EM databases. When this option completes successfully, the previous PostgreSQL database server file system is saved to the following location:
	`<pghome_directory/old_pgsql_<date of operation>`
	If Control-M/EM databases do not exist in the `<pghome_directory>` it is saved in the following location:
	`<db location>/<db name>_old_<date of operation>`
	This option is not enabled if the Control-M/EM database server is running. The following Control-M/EM components must be shut down prior to activating this backup:
	▪ Gateway
	▪ GUI server
	▪ BMC Batch Impact Manager
	▪ Control-M/Forecast
	▪ Global Conditions Server
	▪ Control-M Configuration Management Server
	▪ Control-M Configuration Agent
	Parameters available:
	Restore Directory
	Full path to the directory from which the server file system and Control-M/EM databases should be restored.
	Archive Directory
	Value and location specified in the Archive Directory parameter of the Set Database Archive Mode option.
	When this action is finished successfully, the PosgreSQL database server archive mode switches to off, and the PosgreSQL database server shuts down.
	The specified directory should be exported from the PostgreSQL database server with the same parameter values as the destination PostgreSQL database server.
	This option is only available when using a PostgreSQL database.

Option	Description
Cold Database Restore	Imports the Control-M/EM database schema from the file specified in Backup File parameter of the Cold Database Backup option. This option is not enabled if Control-M database server is running. **Parameters available:** **Restore File** Value and location specified in the Backup File parameter of the Cold Database Backup option. **Administrator password** Password of the database server administrator.

Report Menu options

The following table lists the Report Menu options:

Option	Description
Database Status	Displays various server and client details. ▪ DB Type ▪ Is Up ▪ Is Remote DB ▪ Last Startup Time ▪ DB Server OS Version ▪ DB Server Host Name ▪ DB Server OS Type ▪ DB Server Archive Directory ▪ DB Server Port ▪ DB Client OS Version ▪ DB Client Host Name ▪ DB Client OS Type ▪ DB Server Version ▪ DB Client Version
Database Parameters	Displays the configuration parameters of the server database and the Control-M/EM database client. Configuration parameters are sorted alphabetically.
Database Storage Report	Displays the following attributes of the Control-M/EM database in the server database: ▪ DB Name ▪ Type ▪ Size (refers to the operating system disk space) ▪ Free ▪ Used ▪ Used percentage ▪ Message (Warns the user when there is diminished disk space capacity within the Control-M/EM database server)

Option	Description
Database Version	Displays the general description of the database server. This includes the version number.
List All Active Transactions	List all active transactions of the Control-M/EM database for the database user.

Accessing the Database Utilities Menu using Oracle

This procedure describes how to access the Database Utilities Menu using Oracle.

➢ To access the database Utilities menu using oracle:

- Type the following command at the command prompt:

```
em_database_menu
```

The Database Utilities Menu is displayed:

```
Select one of the following options:
1 - Maintenance
2 - Report
q - quit
Enter option number --->   [q]:
```

The options available in the Database Utilities Menu enable you to access the following sub-menus:

Oracle Maintenance Menu options (on page 499)

Oracle Report Menu options (on page 500)

Within these sub-menus are a variety of functions used to maintain the Control-M database.

Oracle Maintenance Menu options

The following table lists the Oracle Maintenance Menu options:

Option	Description
Build Database	Creates a Control-M/EM schema on the database server.
Maintain Database	Activates statistic procedures in the database server.
Extend database	Extend the database size. Contact your DBA for assistance in performing this action if you are not comfortable performing it yourself.

Oracle Report Menu options

The following table lists the Report Menu options:

Option	Description
Database Status	Displays various server and client details. ▪ DB Type ▪ Is Up ▪ Is Remote DB ▪ Last Startup Time ▪ DB Server OS Version ▪ DB Server Host Name ▪ DB Server OS Type ▪ DB Server Archive Directory ▪ DB Server Port ▪ DB Client OS Version ▪ DB Client Host Name ▪ DB Client OS Type ▪ DB Server Version ▪ DB Client Version
Database Parameters	Displays the configuration parameters of the server database and the Control-M/EM database client. Configuration parameters are sorted alphabetically.
Database Storage Report	Displays the following attributes of the Control-M/EM database in the server database: ▪ DB Name ▪ Type ▪ Size (refers to the operating system disk space) ▪ Free ▪ Used ▪ Used percentage ▪ Message (Warns the user when there is diminished disk space capacity within the Control-M/EM database server)

Option	Description
Database Version	Displays the general description of the database server. This includes the version number.
List All Active Transactions	List all active transactions of the Control-M/EM database for the database user.
Consistency Check	Checks tables in the database. Run this option when you suspect that there is database corruption.

Accessing the Database Utilities Menu using MSSQL

This procedure describes how to access the Database Utilities Menu using MSSQL.

To access the Database Utilities Menu using MSSQL:

- Type the following command at the command prompt:

  ```
  em_database_menu
  ```

  The Database Utilities Menu is displayed:

  ```
  Select one of the following options:
  1 - Maintenance
  2 - Report
  q - quit
  Enter option number --->   [q]:
  ```

  The options available in the Database Utilities Menu enable you to access the following sub-menus:

  - MSSQL Maintenance Menu options (on page 502)
  - MSSQL Report Menu options (on page 503)

  Within these sub-menus are a variety of functions used to maintain the Control-M database.

MSSQL Maintenance Menu options

The following table lists the MSSQL Maintenance Menu options:

Option	Description
Build Database	Creates a Control-M/EM schema on the database server.
Maintain Database	Activates statistic procedures in the database server.
Cold Database Backup	Exports the Control-M/EM database schema to the specified file. The following Control-M/EM components must be shut down prior to activating this backup: ▪ Gateway ▪ GUI server ▪ BMC Batch Impact Manager ▪ Control-M/Forecast ▪ Global Conditions Server ▪ Control-M Configuration Management Server ▪ Control-M Configuration Agent **Parameters available:** **Backup File** Full path to the file into which the database should be backed up. **Administrator password** The Administrator password is not required when using an MSSQL database.
Cold Database Restore	Imports the Control-M/EM database schema from the file specified in Backup File parameter of the Cold Database Backup option. This option is not enabled if Control-M database server is running. **Parameters available:** **Restore File** Value and location specified in the Backup File parameter of the Cold Database Backup option. **Administrator password** The password is not required when using an MSSQL database.
Extend database	Extend the database size. Contact your DBA for assistance in performing this action if you are not comfortable performing it yourself.

MSSQL Report Menu options

The following table lists the MSSQL Report Menu options:

Option	Description
Database Status	Displays various server and client details. ▪ DB Type ▪ Is Up ▪ Is Remote DB ▪ Last Startup Time ▪ DB Server OS Version ▪ DB Server Host Name ▪ DB Server OS Type ▪ DB Server Archive Directory ▪ DB Server Port ▪ DB Client OS Version ▪ DB Client Host Name ▪ DB Client OS Type ▪ DB Server Version ▪ DB Client Version
Database Parameters	Displays the configuration parameters of the server database and the Control-M/EM database client. Configuration parameters are sorted alphabetically.
Database Storage Report	Displays the following attributes of the Control-M/EM database in the server database: ▪ DB Name ▪ Type ▪ Size (refers to the operating system disk space) ▪ Free ▪ Used ▪ Used percentage ▪ Message (Warns the user when there is diminished disk space capacity within the Control-M/EM database server)

Option	Description
Database Version	Displays the general description of the database server. This includes the version number.
List All Active Transactions	List all active transactions of the Control-M/EM database for the database user.
Consistency Check	Checks folders in the database. Run this option when you suspect that there is database corruption.

em_rebuild_database

The em_rebuild_database utility can be invoked to rebuild the Control-M/EM database in the specified PostgreSQL server database.

On Windows, the utility is called **em_rebuild_database.bat.**

If the new DBO is different than the original DBO, you need to use the original DBO to log into Control-M/EM client components until the new DBO is added to the Authorizations list with all required permissions.

To run the em_rebuild database utility, see: Running the em_rebuild_database utility (on page 504)

Running the em_rebuild_database utility

This procedure describes how to run the em_rebuild_database utility, which can be invoked to rebuild the Control-M/EM database in the specified PostgreSQL server database.

➢ To run the em_rebuild_database utility:

1. Do one of the following:
   a. Log on to a Control-M for Databases account (*UNIX*)
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter one of the following commands:
   - UNIX only:

     ```
     em_rebuild_database
     ```

   - Windows only:

     ```
     em_rebuild_database.bat
     ```

3. Follow the prompts in the interactive menu that is displayed.

   ```
   Host Interface Name []:
   Port Number []:
   Database Administrator Password:
   ```

```
Database Owner Login []:
Database Owner Password:
Database Name []:
Database Directory Full Path []:
Encoding [LATIN1]:
Existing Database [N]:
Building the database.
Please confirm  [Y/N]:
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information on the em_rebuild_database utility, see: Management Menu options (on page 506).

Management Menu options

The following table describes the Management Menu options

Option	Description
Host Interface Name	Name of the host computer for the PostgreSQL database server on which Control-M/EM is installed.
Port Number	Communications port on which the PostgreSQL database server listens for queries.
Database Administrator Password	Password of the PostgreSQL database server administrator. The characters you enter are not echoed for security reasons.
Database Owner Login	Name of the Control-M/EM database owner.
Database Owner Password	Password for the Control-M/EM database owner (6 to 30 characters, alphanumeric). The characters you enter are not echoed for security reasons.
Database Name	Name for the Control-M/EM database. This name must be unique within the PostgreSQL database server.
Database Directory Full Path	Location of the Control-M/EM database owner. You must create this location prior to installing the Control-M/EM database. Relevant only for the PostgreSQL database server on UNIX.
Encoding	Language encoding of the Control-M/EM database. Valid values: ▪ LATIN1 (default) ▪ UTF8 If you select UTF8, you need to manually configure the environment settings to enable Control-M/EM components to support this encoding.
Existing Database	Valid values: ▪ **Y** —This value indicates that the Control-M/EM database is defined on a remote PostgreSQL database server. ▪ **N** —This value indicates that the Control-M/EM database is defined on the local PostgreSQL database server.

em_SQL

The em_SQL utility enables you to connect to the database from the command line. This utility is automatically installed on Windows computers during installation of the GUI Server component. To run the em_SQL parameter, see: Running the em_SQL utility (on page 507).

Running the em_SQL utility

This procedure describes how to run the em_SQL utility, which enables you to connect to the database from the command line.

➢ To run the em_SQL utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

   ```
   em_SQL -U <userName> -P <password>
   ```

   SQL, U, and P must be specified in uppercase.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the em_SQL utility see em_SQL utility parameters (on page 507).

em_SQL utility parameters

The following table describes the em_SQL utility parameters:

Parameter	Description
-U <*userName*>	User name. The U must be specified in uppercase.
-P <*password*>	Password. The P must be specified in uppercase.

loader

The loader utility is used to load default data to the Control-M/EM database. This utility should only be used if you ran a clean database operation and want to restore default data to the folders. To run the loader utility see: Running the loader utility (on page 508).

Running the loader utility

This procedure describes how to run the loader utility, which is used to load default data to the Control-M/EM database.

➢ To run the loader utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Change the working directory to the Control-M for Databases home directory.
3. Enter the following command:
   - **UNIX**

     ```
     em loader [-d <directory> | -f <fileName>] [-u <userName> [-p
     <password>] [-db <databaseName>] [-append] [-erase] [-diagon] [-h]
     ```

   - **Windows**

     ```
     loader [-d <directory> | -f <fileName>] [-u <userName> [-p <password>]
     [-db <databaseName>] [-append] [-erase] [-diagon] [-h]
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the loader utility, see: loader utility parameters (on page 509).

loader utility parameters

The following table describes the loader utility parameters:

Field	Description
-d <directory>	The directory in which the files with the default data is located.
-f <*fileName*>	The full path of the file in which the loaded data is saved.
-u <userName>	Database user name. Optional
-p <password>	Database user password. Optional
-db *<databaseName>*	The name of the database to which you want to load the data. Optional
-append	Append the data to the information that already exists.
-erase	Erase the data that already exists.
-diagon	Displays the loader utility information on your screen.
-h	Displays the loader utility usage.

sweep

The sweep utility deletes jobs that are no longer active from the Control-M/EM and Control-M/Server databases. These jobs are referred to as obsolete jobs and are defined in Obsolete criteria (see below).The sweep utility is available in Control-M/EM installations only. To run the sweep utility, see: Running the sweep utility (on page 509).

For reports that are generated by the sweep utility, see: Utility reports (on page 513).

For return codes see: Return codes (on page 514)

To see the criteria used by the sweep utility to determine whether a job or folder is obsolete, see: Obsolete criteria (on page 514).

Running the sweep utility

This procedure describe the sweep utility, which deletes jobs that are no longer active from the Control-M/EM and Control-M/Server databases.

➢ To run the sweep utility:

1. Do one of the following:
   a. Log on to a Control-M for Databases account (*UNIX*)
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

   - (*Windows*)

     ```
     sweep [-U <userName> -P <password> | -PF <passwordFile>]
     -S <serverName> [-Local] [-Date <YYYYMMDD>]
     [-Test | -Sync] [-Timeout <maxSeconds>]
     [-Cyclic <maxFiles>] [-H | -Help] [-Force]
     [-Interval <milliseconds>]
     ```

   - (*UNIX*)

     ```
     em sweep [-U <userName> -P <password> | -PF <passwordFile>] -S
     <serverName> [-Local] [-Date <YYYYMMDD>] [-Test | -Sync] [-Timeout
     <maxSeconds>] [-Cyclic <maxFiles>] [-H | -Help] [-Force] [-Interval
     <milliseconds>]
     ```

     The parameters are described in the following table. The flags are not case sensitive.

3. Press **Enter**.

   The parameters are processed and several reports are generated.

   Avoid updating of the job or folder definitions when the sweep utility is running.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the sweep utility, see: sweep utility parameters (on page 511).

sweep utility parameters

The following table describes the sweep parameters:

Parameter	Description
-U *userName*	Control-M for Databases user name Log on is achieved by providing either a user name and password combination or a password file name.
-P *password*	Control-M for Databases user password

-PF *passwordFile*	Flat file that contains an unencrypted user name and password on separate lines in the following format:
	user=*user_name*
	password=*password*

-S *serverName*	Control-M/EM GUI server logical name
-Local	Deletes all obsolete jobs and folders from the Control-M/EM database only. ▪ You cannot use the -Sync and -Test flags when using -Local. ▪ You can use -Local and -Force to delete obsolete jobs and folders from modified or locked folders in the Control-M/EM database. Log **sweep_sync.txt** is not updated when the utility is activated with -Local. If -Local is not used, the sweep utility behaves as it did before version 6.3.01, deleting jobs from both the Control-M/EM and Control-M/Server databases.
-Date YYYYMMDD date	A date in YYYYMMDD format (for example, 20080915), which sets the selection criteria for obsolete jobs. Default: two days before the current date
-Timeout maxSeconds	Maximum time in seconds to wait for pending callbacks to return with responses to UPLOAD and remote DELETE requests. Default: 900 seconds (15 minutes).
-Test	This flag causes the sweep utility to scan all job definitions and generate the **sweep_obsolete.txt** file, which consists of a report of the current obsolete jobs and folders, without actually deleting the jobs.
-Sync	A flag for synchronizing all non-synchronized folders, listed in the **sweep_sync.txt** file, by rerunning the commands that previously failed, in order to synchronize the Control-M and EM databases.

Parameter	Description
-Cyclic maxFiles	The maximum number of cyclic files for sweep execution log (up to 500 messages each). Default: The execution log is written to one sequential log file without a timestamp.
-H | -Help	Displays the usage.
-Force	This flag causes the sweep utility to scan all folders, including those that are modified or locked, for obsolete jobs. Use -Force only when you are certain there are no definitions in the Control-M/Server database that were not previously downloaded into the Control-M/EM database.
-Interval milliseconds	This flag sets the time to wait between executing the DELETE and UPLOAD commands (in milliseconds). It may be required to avoid time-out problems in Gateway or Control-M. By default, there is no interval.

Utility reports

The reports that are generated by the sweep utility are described in the following table:

File Name	Location	Description
sweep_obsolete.txt	**$HOME/sweep**	Report of all jobs and folders that are candidates for deletion according to the date criteria.
sweep_sync.txt	**$HOME/sweep**	Report of all folders that could not be uploaded to or deleted from the Control-M/Server (either because of a failure or a time-out) and therefore are not currently synchronized.
sweep_log.txt	**$HOME/log**	Execution log (which is cyclic and configurable).

In the **sweep_obsolete.txt** file, the report for obsolete jobs has the following format:

```
delete  job  folder_id  job_id  ctm_name  folder_name  job_name  mem_name
```

In the **sweep_obsolete.txt** file, the report for obsolete folders has the following format:

```
delete  folder  folder_id  ctm_name  folder_name folder_lib
```

The MaxObsoleteJobs system parameter controls the size of the GUI Server memory by limiting the number of obsolete jobs stored in the GUI Server. The default value is 100000.This parameter should only be changed after consulting with BMC Customer Support.

In the **sweep_sync.txt** file, the report for non-synchronized folders has the following format:

```
upload  folder  folder_id  ctm_name  folder_name  folder_lib
```

Only the last versions of the **sweep_obsolete.txt** and **sweep_sync.txt** reports are saved.

sweep utility work flow example

The following example describes typical work flow for the sweep utility:

1. Activate the sweep utility with the –Test parameter to generate the **sweep_obsolete.txt** file, which is essentially a report listing the obsolete jobs and folders.
2. Check the **sweep_obsolete.txt** file and decide if you want to delete the jobs and folders displayed in the report.
3. Activate the sweep utility without the –Test parameter to delete the obsolete jobs and folders from the Control-M/EM and Control-M/Server databases.
4. Check the **sweep_sync.txt** file to find failures that occurred while the sweep utility was attempting to upload or delete folders.
5. Check the **sweep_log.txt** file to understand the reason for the failures and perform the necessary corrections.

6. Activate the sweep utility with the –Sync parameter to upload or delete the folders that were not successfully uploaded or deleted in the previous run.
7. Check the **sweep_sync.txt** file to check that there are no more failures.

Prepare a report of obsolete jobs

The following command specifies that the **sweep_obsolete.txt** file, containing a list of obsolete jobs, is generated according to the obsolete date of September 15, 2008:

```
sweep.exe -U emuser -P empass -S TLVW2K366 -Test -Date 20080915
```

Delete obsolete jobs from the database

The following command specifies that any job from two days ago is considered obsolete and is being deleted from the database:

```
sweep.exe -U emuser -P empass -S TLVW2K366
```

Return codes

The following table describes Sweep utility return codes:

Return code	Description
0	Success (all of the obsolete jobs were deleted, however there may be commands in the sync file that must be run later). If necessary, run the sweep utility again.
1	Failure. Run the sweep utility again.
2	Partial success (some obsolete jobs were not deleted). Run the sweep utility again to delete the remaining jobs or folders.

Obsolete criteria

The following describes the criteria used by the sweep utility to determine whether a job or folder is obsolete.

If the ACTIVE_TILL parameter is specified the following elements are considered as obsolete:

- A regular job, which is not part of a SMART Folder, is considered as obsolete if the following is true:

  ACTIVE_FROM < ACTIVE_TILL and ACTIVE_TILL < obsolete date.

- A RBC is considered as obsolete if:

  ACTIVE_FROM < ACTIVE_TILL and ACTIVE_TILL < obsolete date.

- A SMART Folder is considered as obsolete if:

  All of its RBCs are obsolete (the relationship between the RBCs is always OR).

- A regular folder is considered obsolete if:

  All the jobs in the folder are obsolete.

- A job that is part of a SMART Folder is defined as obsolete if one of the following conditions is true:
    - the SMART Folder is obsolete
    - it satisfies the obsolete criteria according to it own ACTIVE_TILL/ACTIVE_FROM and has no RBCs
    - it has RBCs and is considered as obsolete according to the following table:

Relationship	Job (ACTIVE_TILL/ ACTIVE_FROM)	RBC (ACTIVE_TILL/ ACTIVE_FROM)	Result
AND	Obsolete	Obsolete	Delete job
AND	Obsolete	Active	Delete job
AND	Active	Obsolete	Delete job
AND	Active	Active	Job is active
OR	Obsolete	Obsolete	Delete job
OR	Obsolete	Active	Job is active
OR	Active	Obsolete	Job is active
OR	Active	Active	Job is active
AND	Not defined	Obsolete	Delete job
AND	Not defined	Active	Job is active
OR	Not defined	Obsolete	Job is active
OR	Not defined	Active	Job is active

-

Obsolete criteria examples

This following are examples of how the sweep utility applies the obsolete criteria to determine whether various jobs are obsolete.

The following values apply to all examples in this section:

- SMART Folder G has the following two RBCs:
    - RBC A: ACTIVE_FROM Not defined; ACTIVE_TILL September 12, 2008
    - RBC B: ACTIVE_FROM September 11, 2008; ACTIVE_TILL October 28, 2008

- The obsolete day of the user is September 20, 2008.

For the obsolete day of the user, RBC A is obsolete and RBC B is active, according to the above table.

Job 1 belongs to SMART Folder G and has RBC A.

ACTIVE_FROM is not defined for Job 1.

ACTIVE_TILL is defined for Job 1 for October 1, 2008.

The relationship between Job 1 and its RBCs is AND.

Result: Job 1 is obsolete.

Job 2 belongs to SMART Folder G and has RBC B.

ACTIVE_FROM and ACTIVE_TILL are not defined.

The relationship between Job 2 and its RBCs is OR.

Result: Job 2 is active.

Job 3 belongs to SMART Folder G and has RBCs A, B, and C.

RBC C was deleted from SMART Folder G, but Job 4 still has it.

ACTIVE_FROM and ACTIVE_TILL are not defined.

The relationship between Job 3 and its RBCs is AND.

Result: Job 3 is active, while RBC C is ignored.

General notes

The following notes may be applicable to the features described in this section:

- The sweep utility searches for obsolete jobs and folders on the Control-M/EM side only. It does this by scanning the Control-M/EM database. Any obsolete jobs or folders that are found are deleted from both the Control-M/EM and the Control-M/Server databases. It is therefore recommended that the folders are synchronized in the Control-M/EM and Control-M/Server databases before activating the sweep utility.
- The sweep utility supports only Control-M for z/OS versions 6.1.00 and 6.2.00.
- To delete obsolete folders or jobs a user requires FULL permissions for the specific folder.
- If you want to override any CORBA default parameters for the sweep utility, make the changes under "Sweep" in the CORBA configuration file (**$HOME/etc/domains/config.xml**).

util

util is a multi-purpose utility that can perform the following functions from the command line:

- Export data from the Control-M/EM database
- Import data to the Control-M/EM database
- Delete the Control-M/EM database
- Clear the Control-M/EM database
- Build the Control-M/EM database
- Export a specified definition folder
- Import a specified definition folder
- Import Control-M/Forecast data
- Export Control-M/Forecast data
- Remove Control-M/Forecast data (to save disk space)

To run the util utility, see Running the util utility (on page 518).

When version management mode is selected (default), the following additional functions are available from the command line:

- Export or import of scheduling definitions, either with or without the history of changes made
- Export or import of a specified definition folder, either with or without the history of changes made

The util utility is automatically installed on Windows computers with the Control-M/EM Gateway component. For more information about version management, see *Using Control-M Workload Automation*.

For the DB_ARGS database arguments, see: DB_ARGS database arguments fields (on page 519)

For the functions of the util utility, see: util utility functions (on page 520)

For the function parameters, see: util utility function parameters (on page 524)

To configure the util utility to recognize different delimiters, see: Configure the util utility to recognize different delimiters (on page 528)

DB_ARGS database arguments

All the functions use DB_ARGS in the following format:

```
[-D <database>][{-U <user> -P <password>} |-pf <fileName>]
[-S <server>][-T <level>] [-dbms <system>][-dbtimeout <sec>]
[-dbfile <path>]
```

For Functions and syntax of the util utility (UNIX), add em and a space before specifying util. For example: em util <DB_ARGS> <buildSchema> [-cdbg {1 - 5}][-T 3]

Running the util utility

This procedure describes how to run the util utility, which enables you to perform various function on the Control-M/EM database. for more information, see util (on page 517).

➢ To run the util utility:

1. Open a command prompt window (*Windows*) or go to the command line (*UNIX*).
2. Change the working directory to the Control-M for Databases home directory.
3. Enter one of the following commands:
   - **On Windows**

     ```
     util DB_ARGS function functionParameters
     ```

   - **On UNIX**

     ```
     em util DB_ARGS function <functionParameters>
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

DB_ARGS database arguments fields

The following table describes the DB_ARGS database arguments fields:

Field	Description
-D <database>	Name of database on which to perform the operation. Default: the Control-M/EM database defined during installation
-U <user>	Database user name
-P <password>	Database user password

-pf <*fileName*>	Flat file containing an unencrypted username and password on separate lines in the format:
	user=<*userName* >
	password=<*password*>

-S <server>	Name of database server. Default: database server defined during installation.
-T <level>	Database debug level (Valid values are in the range: 1 - 9).
-dbms <system>	Database management system type. Valid values: Sybase, Oracle If you are using an MSSQL database, specify **Sybase**
-dbtimeout <sec>	Database timeout interval, in seconds.
-dbfile <path>	Full path name for the database debug report. If the file already exists, it is overwritten. Default path and filename: **home_directory/DB_time/date.log**

util utility functions

The following table describes the util utility functions:

Function	Description
-export	Exports job processing definitions, Calendars, Control-M data, and so on, from the Control-M/EM database to an ASCII text file. The data in the mentioned file is separated by field and record delimiters. For more information, see Configuring the util utility to recognize different delimiters below. **Usage:** Specify the following command: util <*DB_ARGS*> -export [-silent] [-cdbg <*debugLevel*>] {-type <all | def | cal | sys |dc | user | alert | gc | maint | collect | view |filter | report | hier | audit | history | bim_forecast | statistics>} {-type net {-name {<*name*>}}} [-file <*file*> | -file - | -dir <*dir*>] [-without_def_history] ■ If the database is exported to a file, you are prompted for a filename. The file that is created is a flat file. ■ If you use FTP to transfer the exported file, use binary mode. ■ If -type all is specified, the job's running information history is not returned. Specify -type history separately if it is required. ■ -type def includes site standard and site customization definitions. ■ If -without_def_history is specified, only the current version of the job's definition is exported. This option is relevant when version management mode is selected (default). For more information, see *Using Control-M Workload Automation.*

<table>
<tr><th>Function</th><th>Description</th></tr>
<tr><td>-import</td><td>Imports job processing definitions, Calendars, Control-M data, and so on, from an ASCII text file to the Control-M/EM database.
The data in the mentioned file is separated by field and record delimiters. For more information, see Configuring the util utility to recognize different delimiters below.
<b>Usage:</b> Specify the following command:
util <DB_ARGS> -import [-silent] [-replace] [-cdbg {debugLevel}] {-type <all | def | cal | sys |dc | user | alert |gc | maint | collect | view |filter | report | hier | audit |history | bim_forecast | statistics>} {-type net {-name {<name>}}} [-file <file> | -file - | -dir <dir> | -dir <fileList>] [-without_def_history]
▪ If the database is imported from a file, you are prompted for a filename. Only .exp files are accepted.
▪ If you use FTP to transfer the imported file, use binary mode.
▪ Stop all Control-M/EM components before doing this operation.
▪ If -type all is specified, the job's running information history is not returned. Specify -type history separately if it is required.
▪ -type def includes site standard and site customization definitions.
▪ If -without_def_history is specified, only the current version of the job's definition is imported. This option is relevant when version management mode is selected (default). For more information, see Version management
Gateway(s) should not be running.
If -replace is specified, your data may be destroyed. Use -replace with extreme caution. Any database element that is replaced is overwritten with data that you supply in the input file. If -replace is not specified and the database attempts to write data that already exists, the utility terminates and a rollback is performed on data written during the write operation.</td></tr>
</table>

Function	Description
-delete	Deletes the specified database folder. **Usage:** Specify the following command: `util<DB_ARGS> -delete [-silent] [-cdbg {debugLevel}]{-name {<name>}}` Stop all Control-M/EM components before doing this operation.
-clean_database	Deletes all database folders. **Usage:** Specify the following command: `util <DB_ARGS> -clean_database [-silent] [-cdbg <debugLevel>]` ▪ Stop all Control-M/EM components before doing this operation. ▪ This utility completely deletes database folders. Use it only with extreme caution. BMC recommends that you backup the database before issuing this command.
-build_schema	Builds the Control-M/EMdatabase. It defines the structure and the type of contents of each data element in the database. **Usage:** Specify the following command: `util <DB_ARGS> -build_schema [-cdbg <debugLevel>]` ▪ Stop all Control-M/EM components before doing this operation. ▪ To load default data to the Control-M/EM database, run the loader utility.

Function	Description
-defexport	Exports the specified definition folder to an ASCII text file. The data in the mentioned file is separated by field and record delimiters. For more information, see Configuring the util utility to recognize different delimiters below. **Usage:** Specify the following command: `util <DB_ARGS> -defexport [-cdbg <debugLevel>] -folder <folder> -dcname <newDc> [-library <newLibrary>] -file <file> [-without_def_history]` ▪ If the database is exported to a file, you are prompted for a file name. ▪ If you use FTP to transfer the exported file, use binary mode. ▪ -type def includes site standard and site customization definitions. ▪ If -without_def_history is specified, only the current version of the definition folder is exported. This option is relevant when version management mode is enabled (default). For more information, see the Using *Control-M Workload Automation*.
-defimport	Imports the specified Control-M for Databases definition folder from an ASCII text file to a specified database folder. The data in the mentioned file is separated by field and record delimiters. For more information, see Configuring the util utility to recognize different delimiters below. **Usage:** Specify the following command: `util <DB_ARGS> -defimport -replace [-cdbg <debugLevel>] [-folder <newFolderName>] [-dcname <newDc>] [-library <newLibrary>] -file <file> [-without_def_history]` ▪ If you use FTP to transfer the imported file, use binary mode. ▪ -type def includes site standard and site customization definitions. ▪ If -without_def_history is specified, only the current version of the definition folder is imported. This option is relevant when version management mode is enabled (default). For more information, see the *Using Control-M Workload Automation*.

util utility function parameters

The following table describes the util utility function parameters:

Parameter	Description	
-silent	Suppresses application messages.	
-replace	Overwrites existing data in the specified folder. Use -replace with extreme caution. Any database element that is replaced is overwritten with data that you supply in the input file. If -replace is not specified and the database attempts to write data that already exists, the utility terminates and a rollback is performed on data written during the write operation.	
-cdbg	Debug level. Range from 1 to 5 (highest level). Use this option only when instructed to do so by BMC Customer Support. Using this option can slow performance and use extra disk space.	
-type	Name of the Control-M for Databases component that is configured by one or more of the following:	
	all	All application data. Default.
	def	Job processing definition data.
	cal	Calendar data, including Rule-Based Calendar entities.
	sys	System data of the application.
	dc	Data center definition data.
	user	Control-M for Databases user data and authorizations.
	alert	Alert data for the Control-M/EM database.
	gc	Global conditions – prerequisite conditions that are passed between Control-M installations by Control-M for Databases.
	maint	Maintenance folders
	collect	Collection definitions
	view	ViewPoint definitions
	filter	Filter definitions
	report	Report definitions

	hier	Hierarchy definitions
	audit	Audit data
	history	History data
	bim_forecast	BMC Batch Impact Manager folders and forecast folders
	statistics	Exceptions data
	More than one -type can be used.	
- type net	Type of Control-M for Databases net.	
	-name	Name of the Control-M for Databases net that is identified by the following parameter:
		name – Name of the net. You can use wildcards when specifying a name: ■ * – represents a string of any length ■ ? – represents one character SSIMU – simulation net SIMU A??????23* – all active nets of netgroup 23
-file	Specifies the details of the source or destination file.	
	file	Name of the source or destination file.
	-file -	■ Exports to standard output. Default: user monitor ■ Imports from standard input. Default: user keyboard
	-dir	Exports or imports from a specified directory.
		dir – Directory of the source or destination file if different from Control-M for Databases home directory.
		file-list – List of text (ASCII) files to be exported or imported, in the format: *filename... filename*
-name	Control-M for Databases net identified by the following name:	
	name	Name of the Control-M for Databases net.
-folder	Folder identified by the following foldername:	
	foldername	Name of the folder.

<table>
<tr><td rowspan="2">-dcname</td><td colspan="2">Control-M identified by the following data center name:</td></tr>
<tr><td>dcname</td><td>Name of the data center.</td></tr>
<tr><td rowspan="2">-library</td><td colspan="2">Type of Control-M for Databases library identified by the following library type:</td></tr>
<tr><td>library</td><td>Type of Control-M for Databases library.</td></tr>
<tr><td>-append</td><td colspan="2">Append the data to the specified folder.</td></tr>
</table>

util utility example

The following are examples of the util utillity:

Export job processing definitions

The following command exports job processing definitions from the default Control-M/EM database to the ASCII data file production for database user dbuser1, whose password is secure01:

```
util -U dbuser1 -P secure01 -export -type def -file production
```

Import calendar data

The following steps are used to import calendar data from the ASCII data file month_cal to the default Control-M/EM database:

Stop all Control-M for Databases gateways.

Specify the following command:

```
util -U dbuser1 -P secure01 -import -type cal -file month_cal
```

Delete database contents

The following command deletes the contents of the folder A0301190CT_BJOB from database CITIES:

```
util -D CITIES -U dbuser1 -P secure01 -delete -name A0301190CT_BJOB
```

Clean the database

The following command cleans database WAGE_RATES:

```
util -D WAGE_RATES -U dbuser1 -P secure01 -clean_database
```

Build a database schema

The following command builds a new schema for database PAYROLL:

```
util -D PAYROLL -U dbuser1 -P secure01 -build_schema
```

Export a database definition folder

The following command exports the INVENTORY definition folder for data center WIP from the default Control-M/EM database to the file wip_stores:

```
util -U dbuser1 -P secure01 -defexport -folder INVENTORY \
-dcname WIP -file wip_stores
```

Import a database definition folder

The following command imports the WORK_IN_PROGRESS definition folder (replacing any data that may have been in this database folder) from file wip_stores to the PRODUCTION database:

```
util -D PRODUCTION -U dbuser1 -P secure01 -defimport \
-replace -folder WORK_IN_PROGRESS -file wip_stores
```

Configure the util utility to recognize different delimiters

During the export process, the util utility reads the fields and records delimiters from the **Defaults.rsc** configuration file or uses default values, if nothing is defined in **Defaults.rsc**. The delimiters are stored in the created export file.

During the import process, the util utility reads the delimiter values from the export file and does not refer to the configuration file.

The default values for the records and fields delimiters are as follows:

- \x1E\x1B\x1F - records delimiter
- \x1C\x1B\x1D - fields delimiter

If you are using either of these sequences in your data, add or modify the following lines in the **Defaults.rsc** file using different values:

```
namevalue * util_exp_records_delimiter <recordsDelimiter>
```

```
namevalue * util_exp_fields_delimiter <fieldsDelimiter>
```

In the following example, the definitions in **Defaults.rsc** are set so that the records delimiter is a sequence of vertical tab, horizontal tab, and escape, and the fields delimiter is defined as the form feed character.

namevalue * util_exp_records_delimiter \x0B\t\x1B

namevalue * util_exp_fields_delimiter \f

The following table shows the characters you can use for the records delimiter and fields delimiter, respectively.

Character code	Description
\n	New line (if it is the only character in records delimiter)
\t	Horizontal tab
\x0B	Vertical tab
\f	Form feed
\x1B	Escape
\x1C	File separator
\x1D	Group separator
\x1E	Record separator
\x1F	Unit separator

The records and fields delimiters must follow these rules:

- a value between 1 and 5 characters.
- The records delimiter must not be a substring of the fields delimiter, and vice versa.

- \0 and blank characters are not valid values. They are not identified and may cause unexpected delimiters to be used.

Control-M/Server utilities

This table lists the Control-M/Server utilities for database maintenance.

Utility Type	Description
ctm_backup_bcp (on page 532)	The ctm_backup_bcp utility exports data from a Control-M/Server database to the **~<controlm_owner>/ctm_server/backup_db** directory.
ctm_restore_bcp (on page 533)	The ctm_restore_bcp utility imports the Control-M/Server database from the bcp_backup directory.
ctmcheckmirror (on page 534)	The ctmcheckmirror utility checks the mirroring of the Control-M/Server database and displays the status.
ctmdbbck (on page 535)	The ctmdbbck utility backs up the Control-M/Server database in the following cases: ▪ Control-M/Server was installed by using a dedicated PostgreSQL database server. Do not use this utility if Control-M/Server was installed with an Oracle database (either existing or dedicated) or an existing PostgreSQL database. ▪ To back up an existing Oracle database or an existing PostgreSQL database, use the ctm_backup_bcp utility (described on ctm_backup_bcp (on page 532)). ▪ Control-M/Server was installed by using either an existing or a dedicated Sybase database server. When working with an existing Sybase database, the ctmdbbck utility must be specified with the device name only, and not with a full path. ▪ Control-M/Server was installed by using an MSSQL database server.
ctmdbcheck (on page 539)	The ctmdbcheck utility displays information about the memory capacity and the status of the Control-M/Server database.
ctmdbmused (on page 543)	The ctmdbmused utility returns the size of a Control-M/Server Master database (*Sybase only*) and the percent used.
ctmdbopt (on page 544)	The ctmdbopt utility calculates database statistics.

Utility Type	Description
ctmdbrst (on page 544)	The ctmdbrst utility restores the Control-M/Server database in the following cases: ▪ Control-M/Server was installed by using a dedicated PostgreSQL database server. Do not use this utility if Control-M/Server was installed with an Oracle database (either existing or dedicated) or an existing PostgreSQL database. ▪ To restore an existing Oracle database or a PostgreSQL database, use the ctm_restore_bcp utility (described on ctm_restore_bcp (on page 533)). ▪ Control-M/Server was installed by using either an existing or a dedicated Sybase database server. ▪ Control-M/Server was installed by using an MSSQL database server.
ctmdbspace (on page 546)	The ctmdbspace utility checks the data and log usage in the Control-M/Server database and displays the usage.
ctmdbtrans (on page 547)	The ctmdbtrans utility lists the active transactions in the database.
ctmdbused (on page 548)	The ctmdbused utility displays the size (in MB), amount, and percentage of current space usage in the Control-M/Server database data and log.
ctmreindex (on page 550)	The ctmreindex utility accesses the Control-M/Server database, reads the data dictionary, reads the index definitions, and then reorganizes indexes. (*UNIX only*)
dbversion (on page 550)	The dbversion utility retrieves the database server version and tests the connectivity of the Control-M/Server database in use.

ctm_backup_bcp

The ctm_backup_bcp utility exports data from a Control-M/Server database to the **~<controlm_owner>/ctm_server/backup_db** directory. Each database folder is backed up as a separate ASCII file. To run the ctm_backup_bcp utility see: Running the ctm_backup_bcp utility (on page 532).

The user running the ctm_backup_bcp utility must have access permission to create the directory **bcp_backup**.

The time taken to backup the Control-M/Server database using the ctm_backup_bcp utility can be shortened by choosing not to backup the Control-M log information (the IOALOG table).

Differences between the ctm_backup_bcp and ctmdbbck utilities

- You can only use the ctm_backup_bcp utility if Control-M/Server is down.
- ctm_backup_bcp exports the data in the Control-M/Server database. The ctmdbbck utility backs up an image of the database for later restoration using ctmdbrst.
- When using ctm_backup_bcp, you cannot specify the backup directory.
- ctm_backup_bcp backs up each database folder to a separate ASCII file. ctmdbbck backs up the entire database to a single binary file.
- For all databases, except PostgreSQL, when using ctmdbbck and ctmdbrst, the restored database must be the same size as the original database. When using ctm_backup_bcp and ctm_restore_bcp, the original and restored databases do not need to be the same size.

The ctmdbbck and ctmdbrst utilities are not relevant for Oracle databases, whether dedicated or existing.

Running the ctm_backup_bcp utility

This procedure describes how to run the ctm_backup_bcp utility, which exports data from a Control-M/Server database to the **~<controlm_owner>/ctm_server/backup_db** directory.

➢ To run the ctm_backup_bcp utility:

1. Shut down Control-M/Server by using the following commands:

   ```
   shut_ca
   ```

   ```
   shut_ctm
   ```

2. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   b. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   c. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

3. Enter the following command:

   ```
   ctm_backup_bcp [-n]
   ```

   -n runs the utility in silent mode. In this mode, confirmation prompt and "backing up contents" messages are not displayed.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details, on the ctm_backup_bcp utility, see: ctm_backup_bcp example (on page 533).

ctm_backup_bcp example

Messages similar to the following examples are displayed:

```
backing up contents of CMS_NODGRP
backing up contents of CMR_AGSTAT
backing up contents of CMS_AGCOMM
backing up contents of CMS_AGSRVTIM
backing up contents of CMR_AJF
backing up contents of CMS_JOBDEF
backing up contents of CMS_USERS
…
Database backup ended successfully.
ctm_backup_bcp -n
```

In this case, Control-M/Server does not display the confirmation prompt and does not issue messages. Only progress dots are displayed.

ctm_restore_bcp

The ctm_restore_bcp utility imports the Control-M/Server database from the bcp_backup directory. The content of this directory was created by the ctm_backup_bcp utility. To run the ctm_backup_bcp utility, see: Running the ctm_restore_bcp utility (on page 534).

Differences between the ctm_restore_bcp and ctmdbrst utilities:

- ctm_restore_bcp imports files created by ctm_backup_bcp. ctmdbrst restores a backup created by ctmdbbck.
- When using ctm_restore_bcp, you cannot specify the directory containing the exported files.
- You can only use ctm_restore_bcp if Control-M/Server is down.
- ctm_restore_bcp imports ASCII files. ctmdbrst restores from a binary file.

```
ctm_restore_bcp
Restoring contents of database.
This procedure DELETES any information in main database
Please confirm [y/n]: y
```

Running the ctm_restore_bcp utility

This procedure describes how to run the ctm_restore_bcp utility, which imports the Control-M/Server database from the bcp_backup directory.

➢ To run the ctm_backup_bcp utility:

1. Shut down Control-M/Server by using the shut_ctm command. Make sure no other users or processes are connected to the SQL Server.
2. Do one of the following:
   a. Log on to a Control-M for Databases account (*UNIX*)
   b. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   c. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
3. Enter the following the command:

```
ctm_restore_bcp [-n]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail on the ctm_restore_bcp utility see: ctm_restore_bcp utility example (on page 534).

ctm_restore_bcp utility example

Messages similar to the following examples are displayed:

```
restoring contents of CMS_NODGRP
restoring contents of CMR_AGSTAT
restoring contents of CMS_AGCOMM
restoring contents of CMS_AGSRVTIM
restoring contents of CMR_AJF
restoring contents of CMS_JOBDEF
restoring contents of CMS_USERS
…
Database restore ended successfully.
ctm_restore_bcp -n
```

In this case, Control-M/Server does not display the confirmation prompt and the "restoring contents" messages. Only progress dots are displayed.

ctmcheckmirror

The ctmcheckmirror utility checks the mirroring of the Control-M/Server database and displays the status.

To run the ctmcheckmirror utility, see:Running the ctmcheckmirror utility (on page 535).

For more information about database mirroring, see Mirroring parameters.

Running the ctmcheckmirror utility

This procedure describes how to run the ctmcheckmirror, which checks the mirroring of the Control-M/Server database and displays the status.

➢ To run the ctm_backup_bcp utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

   ```
   ctmcheckmirror
   ```
3. The ctmcheckmirror utility returns one of the following statuses:
   - Mirroring is enabled.
   - Mirroring is disabled.
   - Mirroring is damaged.

For the Workload Automation parameter name, see Parameter name cross references (on page 34)

ctmdbbck

The ctmdbbck utility backs up the Control-M/Server database in the following cases:

- Control-M/Server was installed by using a dedicated PostgreSQL database server. Do not use this utility if Control-M/Server was installed with an Oracle database (either existing or dedicated) or an existing PostgreSQL database.

  To back up an existing Oracle database or an existing PostgreSQL database, use the ctm_backup_bcp utility (described on ctm_backup_bcp (on page 532)).

- Control-M/Server was installed by using either an existing or a dedicated Sybase database server. When working with an existing Sybase database, the ctmdbbck utility must be specified with the device name only, and not with a full path.
- Control-M/Server was installed by using an MSSQL database server.

For information about Archive mode, see Set Database Archive Mode referred to in the Table in the Section Accessing the Database Utilities Menu using PostgreSQL (on page 492).

When using PostgreSQL, after specifying this utility you are prompted to enter the DBA password to access the Control-M/Server database.

The following describe how to run the utility, backing up the Control-M/Server database and running in silent mode together with an example:

- Running the ctmdbbck utility (on page 536)
- Backing up the Control-M/Server database (on page 537)
- Backing up the Control-M/Server database (PostgreSQL) (on page 537)
- Running the ctmdbbck in silent mode (on page 538)
- ctmdbbck utility example (on page 538)

Running the ctmdbbck utility

This procedure describes how to run the ctmdbbck utility, which backs up the Control-M/Server database.

➢ To run the ctmdbbck utility:

1. Do one of the following:
   a. Log on to a Control-M for Databases account (*UNIX*)
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following the command prompt to use the interactive menu to back up the Control-M/Server database (for both UNIX and Windows):

   ```
   ctmdbbck
   ```
3. To run the ctmdbbck utility not in an interactive mode, run ctmdbbck and specify the following parameters:
   - **For UNIX**

     ```
     ctmdbbck [ -p<administrator password>]
                    [ -f<administrator password file>]
                    [ -d<full path of backup directory/file>]
                    [ -m<backup mode H/C>]
     ```
   - **For Windows**

     ```
     ctmdbbck [ -p<administrator password>]
                      [ -f<administrator password file>]
     ```

```
                    [ -d<full path of backup directory/file>]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Backing up the Control-M/Server database

This procedure describes how to back up the Control-M/Server database (not PostgreSQL)

➢ To back up the Control-M/Server database:

1. Shut-down Control-M/Server and Control-M/Server Configuration Agent before running this utility.
2. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **b.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **c.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
3. Enter the following command:

   ```
   ctmdbbck
   ```

4. Continue the backup by following the prompts.

   For Sybase or MSSQL – You are prompted for the backup device (*<backupDevice>*).

   The device that is specified for this parameter must be either:

   - a valid device defined in an existing Sybase server or MSSQL
   - the full path name of a file to be created by the backup procedure for Sybase or MSSQL

5. If no backup devices exist, or you want to use a new backup device, choose **6 - Add Backup Device** from the Database Maintenance menu.

   If no device is specified for backup of a Sybase database, the default device will be used. BMC Software does not recommend this option.

Backing up the Control-M/Server database (PostgreSQL)

This procedure describes how to back up the Control-M/Server database when using PostgreSQL.

➢ To back up the Control-M/Server database when using PostgreSQL:

1. Shut-down Control-M/Server and Control-M/Server Configuration Agent before running this utility.
2. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **b.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **c.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
3. Enter the following command:

   ```
   ctmdbbck
   ```

4. Continue the backup by following the prompts.

   Prompts and messages similar to the following are displayed:

   ```
   Please Enter Postgres Administrator Password:*******

   Detecting current archive mode ...

   Archive Mode=off

   Done.

   Enter full path destination file name:c:\bckup

   Performing Cold backup into c:\bckup ...

   backup succeeded. It is recommended to check backup file by restoring it
   to test

   environment.

   You can start Control-M/Server.

   The back up is complete.
   ```

Running the ctmdbbck in silent mode

This procedure describes how to run the ctmdbbck utility in silent mode.

➢ To run ctmdbbck in silent mode:

1. Do one of the following:
   - a. Log on to a Control-M for Databases account (*UNIX*)
   - a. Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

   ```
   ctmdbbck -pmanager / -f<passwordFile> -d<backupDirectory> - m<H/C>
   ```

When using PostgreSQL, silent mode is available only on UNIX.

ctmdbbck utility example

The following are examples of the ctmdbbck utility:

Specify the following command to backup the Control-M/Server database where Control-M/Server was installed using an existing Sybase or MSSQL database server and the backup device is controlm_data_bkp:

```
ctmdbbck controlm_data_bkp
```

Specify the following command to back up the Control-M/Server database where Control-M/Server was installed by using a dedicated PostgreSQL database server:

```
ctmdbbck
```

You are prompted for the required parameters.

ctmdbcheck

The ctmdbcheck utility displays information about the memory capacity and the status of the Control-M/Server database.

The ctmdbcheck utility can be run as a cyclic job. To run the ctmdb check utility, see: Running the ctmdbcheck utility (on page 539).

If the *-n* switch is specified in the ctmdbcheck command, only database capacity information is returned, and database thresholds and integrity are not checked.
For performance reasons, run the ctmdbcheck utility during non-peak hours or when Control-M/Server is down. If you need to determine database sizes frequently, use the ctmdbused command. This command displays the size (in MB) of the data and log components of the database plus the amount and percentage of space currently that is used in each component.

Running the ctmdbcheck utility

This procedure describes how to run the ctmdbcheck utility, which displays information about the memory capacity and the status of the Control-M/Server database.

➢ To run the ctmdbcheck utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter one of the following the commands:

- `ctmdbcheck`

  Monitor the database and the transaction log using the ctmdbcheck utility with the following syntax according to the database in use.

- For Oracle specify the following command:

  `ctmdbcheck <generalThreshold%>`

Only data usage may be checked for existing Oracle databases.

- For Sybase enter one of the following commands:
  - `ctmdbcheck [-ddb_threshold %] [-llog_threshold %] [-n]`
  - `ctmdbcheck [general_threshold %]`
- For PostgreSQL, enter the following command:

  `ctmdbcheck <generalThreshold%>`

- For MSSQL enter the following command:

  ```
  ctmdbcheck [-d <dbThreshold%>] [-l <logThreshold%>][-n]
  ```

  ```
  ctmdbcheck 20
  ```

  ```
  db total = 47757170 KB
  ```

```
data used = 6633 KB (0.01%)
Checking database...
Database is OK
```

These commands trigger a Shout message to Control-M/Server if more than the specified percentage of the database or the database transaction log are full. This message can then be used to trigger actions that will extend the appropriate Control-M/Server database component.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmdbcheck utility, see: ctmdbcheck utility parameters (on page 540), ctmdbcheck utility output parameters (on page 541), andctmdbcheck utility examples (on page 541).

ctmdbcheck utility parameters

The following table describes the ctmdbcheck parameters:

Parameter	Description
<dbThreshold%>	Threshold for usage of the Control-M/Server database. If more than the specified percentage of the database is full, a Shout message to Control-M/EM warns that the database should be extended. This variable must be preceded by the -d switch. For example, -d80 indicates that a shout message should be issued if the database is more than 80% full.
<logThreshold%>	Threshold for usage of the transaction log of the Control-M/Server database. If more than the specified percentage of the database is full, a Shout message to Control-M/EM warns that the transaction log should be extended. This variable must be preceded by the -l switch. For example, -l80 indicates that a shout message should be issued if the transaction log is more than 80% full.
<generalThreshold%>	Checks data and log partitions of the Control-M/Server database by the same percentage. For example, if percent usage of either the data area or the transaction log exceeds 80%, ctmdbcheck 80 triggers a Shout message).

ctmdbcheck utility output parameters

The ctmdbcheck utility returns information about the Control-M/Server database. The following table describes the fields that are returned by this utility:

Field	Description
db total	Total amount of memory (in KB) allocated for the database.
data	Total amount of memory (in KB) allocated to the Data partition of the database.
log	Total amount of memory (in KB) allocated to the Log partition of the database.
Data used	Total memory currently used in the Data partition.
Log used	Total memory currently used in the Log partition.

In addition to the above fields, ctmdbcheck also returns one of the following messages describing the current database status:

- Database is OK.
- WARNING: Database is more than half full.
- ATTENTION: Database log segment is more than 90% full.
- ATTENTION: Database is more than 80% full.

ctmdbcheck utility examples

This examples use the ctmdbcheck utility to check database status without specifying any parameters (that is, no Shout messages will be issued for this run on the utility, even if the database is over the desired threshold).

Utility input

```
ctmdbcheck
```

Sybase database – utility output

```
db total = 25000.0 KB (data= 19500.00 , log= 5500.00)

data used = 3696 KB (18%).

log used = 0 KB (0%).

Checking database...

Database is OK.
```

Utility input

```
ctmdbcheck -d80
```

Sybase database – utility output

```
Message 'Warning: DB is more than 80% full', urgency 'U' HostID
'linda'
 - Shout to user 'EM' 'SUCCESS'
db total = 25000.0 KB (data= 19500.00 , log= 5500.00)
data used = 21250 KB (85%).
log used = 0 KB (0%).
Checking database...
Database is OK.
```

This example produces the same display as the first example except that a warning message is generated if the percent usage of the data area is higher than the percentage specified in the command. In this example, the message **data used = 21250 KB (85%)** was generated because 85% exceeds the specified threshold of 80%. This message is also sent to the Control-M/EM alert window.

Utility input

```
ctmdbcheck -l80
```

Sybase database – utility output

```
db total = 25000.0 KB (data= 19500.00 , log= 5500.00)
data used = 21250 KB (85%).
log used = 0 KB (0%).
Checking database...
Database is OK.
```

This command is similar to the second example except that the Log partition is being checked. No warning message is generated because 0% is less than the specified threshold of 80%.

Utility input

```
ctmdbcheck 50
```

The general threshold % option specifies the same percentage for both the database and the log. In this example, if either the database or log exceeds 50%, ctmdbcheck 50 will trigger a shout message.

Utility input

```
ctmdbcheck -d10
```

Oracle 817 database – utility output

```
Folderspace                 Size   % Free
-------------------- -------- -------
RBS                        200M     85%
WIN613O                    150M     95%
WIN613O_INDX                50M     98%
```

Utility input

```
ctmdbcheck -d10
```

Oracle 92 Database – Utility Output

```
Folderspace              Size   % Free
-------------------- -------- --------
CTRLM                    250M      96%
TEMP                     100M      90%
RBS                      300M      99%
```

ctmdbmused utility example

The following example describes a sample output:

```
db master total = 20000.0 KB
data used = 4334 KB (21%)
```

If the data used is more than 80% of the size of the Control-M/Server Master database, the utility output contains the message:

```
CAUTION - DB near capacity. Increase master DB size.
```

If the data used is more than 90%, the utility output contains the message:

```
URGENT - Not enough Master DB free space.
```

BMC recommends running this utility as a cyclic job that automatically generates an appropriate Shout message if the utility output (OUTPUT) contains the phrase "DB near capacity" or "DB free space".

ctmdbmused

The ctmdbmused utility returns the size of a Control-M/Server Master database (*Sybase only*) and the percent used. To run the ctmdbmused utility, see Running the ctmdbmused utility (on page 543).

Running the ctmdbmused utility

This procedure describes how to run the ctmdbmused utility, which returns the size of a Control-M/Server Master database (*Sybase only*) and the percent used.

➢ To run the ctmdbmused utility:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
ctmdbmused
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail on ctmdbmused utility see: ctmdbmused utility example (on page 543).

ctmdbopt

The ctmdbopt utility calculates database statistics. This utility wraps the relevant database packages that collect statistics on all Control-M/Server database folders. You can run this utility while Control-M/Server is running.

The ctmdbopt utility collects folder and index statistics on the Control-M schema by using the DBMS_STATS package. It affects all Control-M/Server for UNIX and Microsoft Windows database folders. The statistics helps the optimizer to choose the fastest way to retrieve data (full folder scan, index scan, or any other way).

Gathering statistics improves database performance. BMC recommends that you run this utility on a daily basis, so that the database optimizer has updated statistics. To run the ctmdbopt command, see: Running the ctmdbopt utility (on page 544).

Running the ctmdbopt utility

This procedure describes how to run the ctmdbopt utility, which calculates database statistics.

➢ To run the ctmdbopt utiliity:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter one of the following the command:

```
ctmdbopt
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

ctmdbrst

The ctmdbrst utility restores the Control-M/Server database in the following cases:

- Control-M/Server was installed by using a dedicated PostgreSQL database server. Do not use this utility if Control-M/Server was installed with an Oracle database (either existing or dedicated) or an existing PostgreSQL database.

  To restore an existing Oracle database or a PostgreSQL database, use the ctm_restore_bcp utility (described on ctm_restore_bcp (on page 533)).
- Control-M/Server was installed by using either an existing or a dedicated Sybase database server.
- Control-M/Server was installed by using an MSSQL database server.

For more information about backup types, see Database operation and maintenance.

To run the ctmbrst utility, see Running the ctmbrst utility (on page 545) and to run in silient mode see: Running the ctmdbrst in silent mode (on page 546). To restore the Control-M/Server database, see: Restoring the Control-M/Server database (on page 545).

Running the ctmbrst utility

This procedure describes how to run the ctmdbrst utility, which restores the Control-M/Server database.

➢ To run the ctmdbrst utility:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Do one of the following:

   - To invoke the ctmdbrst utility for Control-M/Server installed with Sybase or MSSQL, enter the following command:

     ```
     ctmdbrst
                        [ -f<administrator password file>]
                     [ -d<full path of backup directory/file>]
                     [ -m  H/C]
     ```

   - To invoke the ctmdbrst utility for Control-M/Server installed with PostgreSQL, run the following command (this mode is UNIX only):

     ```
     ctmdbrst [ -p<administrator password>]
                    [ -f<administrator password file>]
                    [ -d<full path of backup directory/file>]
                    [ -a<full path of archive directory>]
                [ -m<restore mode H/C>]
     ```

   - Enter the following command at the command prompt to use the interactive menu (this mode is Windows only):

     ```
     ctmdbrst
     ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Restoring the Control-M/Server database

This procedure explains how to restore the Control-M/Server database by using the ctmdbrst utility.

➢ To restore the Control-M/Server database:

1. Shut down Control-M/Server and the Configuration Agent before invoking this utility. Make sure that no other users or processes are connected to the SQL server.

2. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   b. Open a command prompt window (*Windows*) where Control-M/EM is installed.

  c. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

3. Enter the following:

   ```
   ctmdbrst
   ```

   Continue the restore procedure by following the prompts.

For Sybase or MSSQL – you are prompted for the backup device (*<backupDevice>*). This device was used to back up the Control-M/Server database. The device specified for this parameter must be either a valid device defined in Sybase, or the full path name of a file to be used as input for the ctmdbrst utility.

If no device is specified for restoration of a Sybase database, the default device (tapedump2) will be used. This option is not recommended.

Running the ctmdbrst in silent mode

This procedure describes how to run the ctmdbrst utility in silent mode.

➢ To run ctmdbrst in silent mode:

1. Do one of the following:

   a. Log on to a Control-M for Databases account (*UNIX*)

   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

   ```
   ctmdbrst -pmanager -d<restoreDirector>  -m<H/C> -a<archive Director>
   ```

   The following command causes the Control-M/Server database to be restored from the default backup device:

   ```
   ctmdbrst
   ```

ctmdbspace

The ctmdbspace utility checks the data and log usage in the Control-M/Server database and displays the usage. The utility returns a "failed" status if the usage exceeds the specified limit. To run the ctmdbspace, see: Running the ctmdbspace utility (on page 546).

ctmdbspace can be included in the Control-M Watchdog process. For more information, see Watchdog process parameters.

Running the ctmdbspace utility

This procedure describes how to run the ctmdspace utility, which checks the data and log usage in the Control-M/Server database and displays the usage.

➢ To run the ctmdbspace utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
ctmdbspace -limit <amount> [-quiet]
```

The *<amount>* variable is the maximum amount (percentage) of data and log usage in the database. Use the optional -quiet parameter to suppress the display.

The following command returns a "failed" status if Control-M/Server database usage is more than 50%:

```
ctmdbspace -limit 50%
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

ctmdbtrans

The ctmdbtrans utility lists the active transactions in the database. A transaction is defined as the unit of work performed by Control-M in the database. Each transaction is assigned a unique name identifying that specific unit of work.

You may be asked by technical support to run this utility and to provide them with the output for debugging purposes.

Another way to list active transactions in the database, is to select "List Active Transactions" from the Troubleshooting menu. To run the ctmdbtrans utility and to run in silent mode, see: Running the ctmdbtrans utility (on page 547) and Running the ctmdbtrans utility in batch mode (on page 548)

For Sybase databases, after specifying the ctmdbtrans utility, you are prompted to enter the DBA password to access the Control-M/Server database.

Running the ctmdbtrans utility

This procedure describes the ctmdbtrans utility, which lists the active transactions in the database.

➢ To run the ctmdbtrans utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command to see which transactions are active in the database:

```
ctmdbtrans
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Running the ctmdbtrans utility in batch mode

This procedure describes how to run the ctmdbtrans utility in batch mode.

➢ To run the ctmdbtrans utility in batch mode:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Do one of the following:
   - For Sybase databases specify the DBA password in a file, and enter the following command:

     ```
     ctmdbtrans < <fullPathFileName>
     ```

     `<fullPathFileName>` is the full path to the file in which the DBA password is specified.
   - For PostgreSQL, Oracle and MSSQL databases, enter the following command:

     ```
     ctmdbtrans
     ```
   - For UNIX machines you can refresh transactions every given amount of time by specifying the sleep time parameter by entering the following command:

     ```
     ctmdbtrans
     ctmdbtrans n (number in seconds)
     ```

ctmdbused

The ctmdbused utility displays the size (in MB), amount, and percentage of current space usage in the Control-M/Server database data and log. To run a ctmbused utility see: Running the ctmbused utility (on page 548).

Running the ctmbused utility

This procedure describes how to run the ctmdbused utility displays the size (in MB), amount, and percentage of current space usage in the Control-M/Server database data and log.

➢ **To run the ctmbused utility:**

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

   ```
   ctmdbused
   ```

Use this utility if you need to determine Control-M/Server database sizes frequently.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on running the ctmbused utility, see ctmbused utility example (on page 549).

ctmbused utility example

These examples use the ctmdbused utility to display Control-M/Server database data and log size, amount of current usage and percentage of space, using Sybase and MSSQL databases.

Utility input

```
ctmdbused
```

Utility output

```
db total = 45000.0 MB (data= 3000.00 , log= 15000.00)

data used = 4174 MB (13%).

log used = 40.00 MB (0%)
```

This example uses the ctmdbused utility to display Control-M/Server database data and log size, amount of current usage and percentage of space, using Oracle 11.

Utility input

```
ctmdbused
```

Oracle 11 database – utility output

```
db total = 563200.0 MB

data used = 10048 MB (4%).
```

This example uses the ctmdbused utility to display Control-M/Server database data, amount of current usage and percentage of space, using a PostgreSQL database.

Utility input

```
ctmdbused
```

Utility output

```
db total = 50404551 MB

data used = 863570 KB (1.71%)
```

ctmreindex

The ctmreindex utility accesses the Control-M/Server database, reads the data dictionary, reads the index definitions, and then reorganizes indexes. (*UNIX only*)

This utility is relevant only on Control-M/Server with a Sybase database.

The Control-M/Server Sybase database must be running so that the required index operations can be performed. However, this utility should be run only when database activity is low.

This utility enables Control-M/Server database queries to execute faster by ensuring that indexes are well-balanced. For more information about indexes, refer to Sybase manuals. To run the ctmreindex utility, see: Running the ctmreindex utility (on page 550).

Running the ctmreindex utility

This procedure describes how to run the ctmreindex, which accesses the Control-M/Server database, reads the data dictionary, reads the index definitions, and then reorganizes indexes (*UNIX only*).

➢ To run the ctmreindex utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the following command:

```
ctmreindex
```

The following command causes the ctmreindex utility to reorganize indexes in the Control-M/Server Sybase database:

```
        ctmreindex
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

dbversion

The dbversion utility retrieves the database server version and tests the connectivity of the Control-M/Server database in use. The database can be any one of the following database servers: Oracle, Sybase, MSSQL, or PostgreSQL. To run the dbversion utility, see: Running the dbversion utility (on page 550).

Running the dbversion utility

This procedure describes how to run the dbversion utility, which retrieves the database server version and tests the connectivity of the Control-M/Server database in use.

➢ To run the ctmbused utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
dbversion
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Interactive database utilities

The utilities discussed in this section can be invoked interactively to facilitate day-to-day maintenance and diagnostics of Control-M/Server running with either PostgreSQL, MSSQL or Oracle databases.

The interactive database utilities are located at the following path for Windows:

```
<full path of the Control-M/EM exe directory/DBUtils>
```

For details about Return codes, see: Return codes (on page 552).

The following interactive database utilities are listed:

- Running the DBUColdBackup utility (on page 552)
- Running the DBUColdRestore utility (on page 553)
- Running the DBUStart utility (on page 554)
- DBUStop (on page 555)
- Running the DBUVersion utility (on page 556)
- DBUStatus (on page 558)
- DBUStorage (on page 560)
- Running the DBUTransactions utility (on page 561)
- Running the DBUShow utility (on page 562)

Return codes

The return codes listed in the following table are issued by the interactive database utilities.

Return code	Description
0	The action completed successfully.
1	The action failed and an error message is issued. Error messages have the following format: `<Module number> - <Module name>: <Error Code (numerical)> - <Error description>` AP-7 - Permission Module: 2 - This utility could not be used with the existing PostgreSQL database.
2	The action completed successfully. A warning message was displayed, although this warning had no effect on the action itself.

Running the DBUColdBackup utility

This procedure describes how to run the DBUColdBackup utility, which enables you to export the Control-M/EM database schema to the specified file.

➢ To run the DBUColdBackup utility:

- Type the following command:

```
DBUColdBackup
[ -TRACE_LEVEL <error|log|info> ]
[ -HELP ]
[ -BACKUP_FILE <Full Path of Backup File Name> ]
[ -ADMINISTRATOR_PASSWORD <Administrator Password> ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on DBUColdBackup, see: DBUColdBackup utility parameters (on page 553) and DBUColdBackup utility example (on page 553).

DBUColdBackup utility parameters

The following table describes the DBUColdBackup utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.
-BACKUP_FILE	Full path to the file into which the database should be backed up. Mandatory.
-ADMINISTRATOR_PASSWORD	Password of the database server administrator. Mandatory.

DBUColdBackup utility example

The following example describes a DBUColdBackup utility sample output:

```
database was backup to /home1/ctm640pg/Backup.bck
```

Running the DBUColdRestore utility

This procedure describes how to run the DBUColdRestore utiliy, which imports the Control-M/EM database schema from the file specified in the BACKUP_FILE parameter of the DBUColdBackup utility.

➢ To run the DBUColdRestore utility:

- Type the following command:

  ```
  DBUColdRestore
  [ -TRACE_LEVEL <error|log|info> ]
  [ -HELP ]
  [ -RESTORE_FILE <Full Path of Restore File Name> ]
  [ -ADMINISTRATOR_PASSWORD <Administrator Password> ]
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details about the DBUColdRestore utility, see: DBUColdRestore utility parameters (on page 554) and DBUColdRestore utility example (on page 554).

DBUColdRestore utility parameters

The following table describes the DBUColdRestore utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.
-RESTORE_FILE	Value and location specified in the BACKUP_FILE parameter of the DBUColdBackup utility. Mandatory. The source and destination databases should have the same encoding settings configured.
-ADMINISTRATOR_PASSWORD	Password of the database server administrator. Mandatory.

DBUColdRestore utility example

The following example describes a utility DBUColdRestore sample output:

```
restore completed
```

Running the DBUStart utility

This procedure describes how to run the DBUStart utility, which enables you to start the database server and the related services. When invoking this utility with a PostgreSQL database, the utility is only enabled on Control-M/EM running with a dedicated PostgreSQL database server.

If this option is invoked on Control-M/EM running with an existing PostgreSQL database, an error message is displayed.

➢ To run the DBUStart utility:

- Type the following command:

  ```
  DBUStart
  [ -TRACE_LEVEL <error|log|info> ]
  [ -HELP ]
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information about the DBUStart utility, see: DBUStart utility parameters (on page 555) and DBUStart utility example (on page 555).

DBUStart utility parameters

The following table describes the DBUStart utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.

DBUStart utility example

The following example describes a DBUStart utility sample output:

```
PostgreSQL server started
```

If the database server has already been started, the utility returns a "failed" status and a message similar to the following is issued:

sh-500 - sh-500 module : 500 - Server is already up

DBUStop

This procedure describes how to run the DBUStop utility, which enables you to stop the database server and the related services. When invoking this utility with a PostgreSQL database, the utility is only enabled on Control-M/EM running with a dedicated PostgreSQL database server.

If this option is invoked on Control-M/EM running with an existing PostgreSQL database, an error message is displayed.

➢ To run the DBUStop utility:

- Type the following command:

```
DBUStop
[ -TRACE_LEVEL <error|log|info> ]
[ -HELP ]
[ -FORCE <Y|N> ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information on DBUStop utility, see: DBUStop utility parameters (on page 556) and DBUStop utility example (on page 556).

DBUStop utility parameters

The following table describes the DBUStop utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.
-FORCE	Enables the database server processes and listener to abort. **Valid values:** ▪ Y ▪ N (default)

DBUStop utility example

The following example describes a DBUStop utility sample output:

```
PostgreSQL server stopped
```

Running the DBUVersion utility

This procedure describes how to run the DBUVersion utility, which enables you to displays the general description of the database server, including the version number.

➢ To run the DBUVersion utility:

- Type the following command:

```
DBUVersion
[ -TRACE_LEVEL <error|log|info> ]
[ -HELP ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more detail about the DBUVersion utility, see: DBUVersion utility parameters (on page 557) and DBUVersion utility example (on page 557).

DBUVersion utility parameters

The following table describes the DBUVersion utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.

DBUVersion utility example

The folllowing example describes a DBUVersion utility sample output:

```
PostgreSQL 8.2.4 on sparc-sun-solaris2.8, compiled by /opt/SUNWspro/bin/cc
-xtarget=ultra -xarch=v9 -xO5 –Xa
```

DBUStatus

The DBUStatus utility displays database client details for all supported databases.

- DB Type
- Is Up
- Is Remote DB

Last Startup Time

- DB Server OS Version
- DB Server Host Name
- DB Server OS Type
- DB Server Archive Directory
- DB Server Port
- DB Client OS Version
- DB Client Host Name
- DB Client OS Type
- Number of Connections
- Number of Backend Processes
- DB Server Version
- DB Client Version

To run the DBUStatus utility, see Running the DBUStatus utility (on page 558).

Running the DBUStatus utility

This procedure describes how to run the DBUStatus utility, which enables you to display database client details for all supported databases.

➢ To run the DBUStatus utility:

- Type the following command:

```
DBUStatus
[ -TRACE_LEVEL <error|log|info> ]
[ -HELP ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the DBUStatus utility, see: DBUStatus utility parameters (on page 559) and DBUStatus utility example (on page 559).

DBUStatus utility parameters

This table describes the DBUStatus utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.

DBUStatus utility example

The following example describes a DBUStatus utility sample output:

```
DB=PostgreSQL

Current DB status=Up

Up Time= 2008-06-04 13:41:03.114827+03

Is DB Remote=false

Server Host Name=cyborg

Server Host Version=SunOS cyborg 5.10 Generic_120011-14 sun4u sparc
SUNW,Sun-Fire-V440

Client Host Name=cyborg

Client Host Version=SunOS cyborg 5.10 Generic_120011-14 sun4u sparc
SUNW,Sun-Fire-V440

DB Client Version=8.2.4

DB Server Version=8.2.4

Connections To DB= 1

Port=5432

Archive Mode=off
```

DBUStorage

The DBUStorage utility displays the following attributes of Control-M/EM for all supported databases:

- DB Name
- Type
- Size - refers to the operating system disk space
- Free
- Used
- Used percentage
- Location
- Message - Warns the user when there is diminished disk space capacity within the Control-M/EM database server
- Recommendation

Running the DBUStorage utility

This procedure describes how to run the DBUStorage utility, which enable you to display various attributes of Control-M/EM for all supported databases. For more information, see DBUStorage (on page 560).

➢ To run the DBUStorage utility:

- Type the following command:

```
[ -TRACE_LEVEL <error|log|info> ]
[ -HELP ]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details about the DBUStorage utility, see: DBUStorage utility parameters (on page 561) and DBUStorage utility example (on page 561)

DBUStorage utility parameters

This table describes the DBUStorage utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.

DBUStorage utility example

The following example describes a DBUStorage utility sample output:

```
sahara-ctm700pg [1] DBUStorage
DB name = ctrlm700
Type = data+log
Size = 17171 MB
 Free = 17163 MB
Used = 7 MB
Used_percentage = 0.05%
Location = /home/ctm700pg/pgsql/app_data/ctrlm700
Message = none
Recommendation = none
```

Running the DBUTransactions utility

This procedure describes how to run DBUTransactions utility, which enables you to list all active transactions of the Control-M/EM database.

➢ To run the DBUTransactions utility:

- Type the following command

  ```
  DBUTransactions
  [ -TRACE_LEVEL <error|log|info> ]
  [ -HELP ]
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

DBUtransactions utility parameters

This table describes the DBUtransactions utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.

DBUtransactions utility example

The following example describes a DBUtransactions utility sample output:

Sample output

```
number of connections = 2

connection 1 db_name=ctrlm640 os_proc=1024172 user_name=ctmuser
query_start_time=2008-07-09 07:03:59.125859 client_ip=137.72.205.101

connection 2 db_name=ctrlm640 os_proc=639076 user_name=ctmuser
query_start_time=2008-07-09 07:27:53.37908 client_ip=137.72.205.101

number of transactions = 1

transactions 1 db_name=ctrlm640 os_proc=1011810 user_name=ctmuser
query_start_time=2008-07-09 07:28:09.797397 client_ip=137.72.205.101
current_transaction=select dbu_transactions('12155776886200000000')

number of locks = 0
```

Running the DBUShow utility

This procedure describes how to run the DBUShow utility, which enables you to display the configuration parameters of all supported databases and the Control-M/EM database client.

Configuration parameters are sorted alphabetically.

➢ To run the DBUShow utility:

- Type the following command:

  ```
  DBUShow

  [ -TRACE_LEVEL <error|log|info> ]

  [ -HELP ]
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the DBUShow utility, see DBUShow utility parameters (on page 563) and DBUShow utility example (on page 563).

DBUShow utility parameters

This table describes the DBUShow utility parameters:

Parameter	Description
-TRACE_LEVEL	Trace level. Valid values: error, log, info. **Default:** error. Optional. Use this option *only when instructed to do so* by BMC Customer Support. Using this option can slow performance and use extra disk space.
-HELP	Displays the usage, then exits with **success** status.

DBUShow utility example

The following example describes a DBUShow utility sample output:

Following configuration parameters are issued and more:

```
add_missing_from=off source=default
allow_system_folder_mods=off source=default
archive_command= source=configuration file
archive_timeout=0 source=default
array_nulls=on source=default
authentication_timeout=60 source=default
autovacuum=on source=configuration file
autovacuum_analyze_scale_factor=0.1 source=default
autovacuum_analyze_threshold=250 source=default
autovacuum_freeze_max_age=200000000 source=default
autovacuum_naptime=60 source=default
autovacuum_vacuum_cost_delay=-1 source=default
autovacuum_vacuum_cost_limit=-1 source=default
autovacuum_vacuum_scale_factor=0.2 source=default
autovacuum_vacuum_threshold=500 source=default
```

Security

The security utilities define the security authorizations for various users and Control-M components, including:

- User registration and password changes
- Access to databases and other components

By including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utilities used to define security authorizations for various users and Control-M components

Utility	Description
ctmcpt (on page 566)	Registers a user in the Control-M/Server registry database.
ctmpasswd (on page 567)	Enables you to change the Control-M/Server User's password for accessing the database.
ctmpwd (on page 591)	(*Windows only*) Create and modify Control-M/Agent users and passwords.
ctmsec (on page 568)	Protects Control-M against unauthorized usage or modification.
ctmsetown (on page 587)	Manages the details of Control-M/Agent and remote host users.
emcryptocli (on page 565)	Creates an encrypted version of the password you submit.

Control-M/EM utilities

Control-M/EM utility for security:

- emcryptocli (on page 565)

emcryptocli

The emcryptocli utility creates an encrypted version of the password you submit. To run the emcryptocli utility, see Running the emcryptocli utility (on page 565).

If the Control-M for Databases administrator user name or password is changed in the Control-M/EM database, it must also be updated manually in all relevant **mcs.ini** files. By default, the password is encrypted in the **mcs.ini** file. Use the emcryptocli utility to generate the encrypted version of your new password.

Usage: You can invoke emcryptocli in either of the following modes:

- **Trial mode:** You submit the new password and emcryptocli creates an output text file in the specified location containing the encrypted version of that password. You can copy the encrypted text to appropriate places in the **mcs.ini** file manually.
- **Operational mode:** When you submit your username and new password, emcryptocli creates an encrypted version of the password and inserts it in the appropriate places in the text of the **mcs.ini** file. **mcs.ini** is saved automatically.

Running the emcryptocli utility

This procedure describes how to run the emcryptocli utility in trial or operation mode, which creates an encrypted version of the password you submit.

➢ To run the ctmjsa utility in trial mode:

1. Do one of the following:

  a. Log on to a Control-M for Databases account (*UNIX*)

  a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

  b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Change the working directory to Ini under the Control-M for Databases home directory.

3. Type one of the following commands:

- For trial mode: emcryptocli.exe <*newPassword*> <*outputFileName*>
- For Operation mode: `emcryptocli.exe <userName> <newPassword> mcs.ini <pathName>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information, see emcryptocli utility parameters (on page 566).

emcryptocli utility parameters

The following table describes the emcryptocli utility trial and operation mode parameters:

Item	Description
<*newPassword*>	New Control-M for Databases administrator password.
<*outputFileName*>	(Trial Mode only) Full path name of the output file created by emcryptocli. The file contains the encrypted version of the password that was submitted.
<*userName*>	(Operation Mode only) Control-M for Databases administrator user name.
mcs.ini <*pathName*>	(Operation Mode only) Full path name of the **mcs.ini** file (for example, **windir\system32\mcs.ini**).

Control-M/Server utilities

This table lists the Control-M/Server utilities for security.

Utility Type	Description
ctmcpt (on page 566)	The ctmcpt utility registers a user in the Control-M/Server registry database.
ctmpasswd (on page 567)	The ctmpasswd utility enables the administrator to change the Control-M/Server User's password for accessing the database.
ctmsec (on page 568)	The ctmsec utility can be invoked in interactive or batch mode.
ctmsetown (on page 587)	The ctmsetown command line utility manages the authentication credentials of job owners for both local and agentless jobs.

ctmcpt

The ctmcpt utility registers a user in the Control-M/Server registry database. The Shout to E-Mail facility requires that Control-M/Server be running as either a service or a program under the account of a user who is registered in the Control-M/Server registry database (*Windows only*). To run the ctmcpt utility, see: Running the ctmcpt utility (on page 567).

Running the ctmcpt utility

The following procedure describes how to run the ctmcpt utility which registers a user in the Control-M/Server registry database.

➢ To run the ctmcpt utility:

1. Do one of the following:

a. Open a command prompt window where Control-M/EM is installed.

a. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. To register a user for the first time enter the following command:

```
ctmcpt <username> "" <password>
```

The quotation marks "" indicates the absence of an old password.

3. To change the password for a user who is already registered, enter the following command:

```
ctmcpt <username> <old_password> <new_password>
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmcpt utility, see: ctmcpt utility example (on page 567).

ctmcpt utility example

The following examples describe the ctmcpt utility:

The following command registers new user user1 in the Control-M/Server registry database with password *pass01*:

```
ctmcpt user1 "" pass01
```

The following command changes the password for user1 from *pass01* to *do2day*:

```
ctmcpt user1 pass01 do2day
```

ctmpasswd

The ctmpasswd utility enables the administrator to change the Control-M/Server User’s password for accessing the database. Only an administrator can change the password. To run the ctmpassword utility, see Running the ctmpasswd utility (on page 567).

Running the ctmpasswd utility

This procedure describes how to run the ctmpasswd utility**,** which enables the administrator to change the Control-M/Server User’s password for accessing the database.

➢ To run the ctmpasswd utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Change the working directory to Ini under the Control-M for Databases home directory.

3. Enter the old password for the Control-M/Server account.

4. Enter the new password.

   Ensure that the password contains at least 6 characters.

5. Re-enter the new password.

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

ctmsec

The ctmsec utility can be invoked in interactive or batch mode. For more information about Control-M security concepts, see Control-M security .

The ctmsec utility is used to:

- Add, delete, or modify specific users in the Control-M Security database
- Add, delete, or modify specific groups in the Control-M Security database
- Assign authorizations to a user or group to perform actions on a Folder
- Assigns authorizations to a user or group to perform actions relating to Control-M entities

The following topics are discussed in this section:

- Security considerations (on page 569)
- Security maintenance utility (Interactive mode) (on page 570)
- Security maintenance utility (Batch mode) (on page 584)
- Exporting security definition folders (on page 586)
- Importing security definition folders (on page 586)

For the Workload Automation parameter name, see Parameter name cross references (on page 34).

Security considerations

Control-M/Server includes security features that protect Control-M against unauthorized usage or modification. These features enhance the standard UNIX and Windows security, and provides an additional application-level security layer.

Using Control-M security, you can specify actions that each Control-M/EM user or Control-M/Server user is authorized to perform. These authorizations are used to perform security checks each time one of the following actions is attempted:

- Accessing a Folder (to add, delete, or modify a job definition).
- Ordering a job.
- Selecting and submitting a job.
- Commands affecting jobs in Active Jobs database (for example, Hold, Confirm, Rerun).
- Maintenance of Control-M entities (for example, calendars, prerequisite conditions).

Security verifications for the above actions are implemented according to the specifications in a database of authorizations. This database can be modified by the security officer or systems manager to meet the needs of the enterprise. For more information, see Security maintenance utility (Interactive mode) (on page 570)

Level of application security

Control-M provides the following levels of application security for users not explicitly defined in the Control-M Security database:

Security level	Description
Restricted	A user not defined in the Control-M Security database is regarded as having no authorizations and cannot perform any function requiring security authorization.
Unrestricted	A user not defined in the Control-M Security database is regarded as having all Control-M application authorizations.

Regardless of which level is implemented:

- A user, for whom one or more authorizations have been assigned in the Security database, can only perform those actions.
- The owner of each job processing definition must be defined as a user on the agent computer. Otherwise, Control-M/Agent will not execute the job.

The security level is determined by the value of the Control-M system parameter Full Security.

Encryption and compression

- If Control-M Option for SSL is installed, Secure Sockets Layer encryption and compression provide security for Control-M/Server communication with Control-M/EM and Control-M/Agents. For more information, see the SSL Management.

- When working with the Control-M/Server Security facility, wildcard characters are available for all options. Wildcard characters * and $ are translated during runtime security checking. (For example, if User1 is granted full Folder authorization for folder ACC*, Control-M allows User1 to update or order any folder whose name starts with ACC). Valid wildcard characters:

  * represents any number of characters (including none).

  $ represents a single character.

  Wildcard character authorizations do not override full name authorizations. (For example, if User1 from the example above is also defined to have only Read privileges for ACC999, Control-M will not allow User1 to update or order folder ACC999).

Security maintenance utility (Interactive mode)

The ctmsec Control-M Security Maintenance utility defines users in the Control-M Security database and assigns authorizations required for working with Control-M using the Control-M Configuration Manager. ctmsec runs on the Control-M/Server computer.

Changes made by this utility are implemented only after you exit the utility.

Users can be defined as part of a group. Authorizations can be specified for a specific user, for a group, or for both. See Security maintenance utility (Batch mode) (on page 584).

When assigning a user to a group, the following rules apply:

- If there are no authorizations defined for the user, the user inherits the authorizations for the group.
- If there are authorizations defined for a user, these authorizations take precedence.
- When defining an authorization for a user (for example, Folder), use of the (D)efault setting enables the specific authorization (for example, Read) defined for the group.
- If all of a user's authorizations for a specific Control-M element (for example, Folder) are defined with a (D)efault setting, the user's authorizations for that element can be deleted more efficiently.
- Authorizations not specifically defined for a group, or for a user not belonging to a group, revert to the Full Security parameter setting. See ctmsys (on page 445)

Certain functions of the ctmsec utility can be activated directly from a command line. For more information, see Security maintenance utility (Batch mode) (on page 584). In addition, certain functions of the ctmsec utility can be activated using the Control-M Configuration Manager. For more information, see Control-M security.

The security of Sub-folders and jobs within Sub-folders is determined according the security that is set for SMART folders.

Displaying the Control-M Security Maintenance Utility

This procedure describes how to display the Control-M Security utility.

➢ To display the Control-M Security Maintenance utility:

1. Log on to the Control-M/Server computer as the Control-M/EM owner (for example, user controlm).
2. Do one of the following:
   - a. Specify the ctmsec command.

b. Choose **Security Authorization => Security Maintenance Utility** from the Control-M/Server Main Menu. For more information, about the Control-M Main Menu options, see Control-M security.

The **Security Maintenance Menu** appears.

User maintenance

The User Maintenance option of the ctmsec utility is used to add, delete, or modify specific users in the Control-M Security database. You can select Security Maintenance Main Menu by selecting option 1.

- Each Control-M/EM user who performs actions affecting the Control-M/Server database or jobs in the Active Jobs database must be defined in the Control-M Security database when full security is on. In addition, all other users who invoke Control-M Security utilities must be defined in the Security database and assigned appropriate privileges.
- If the user in the commands listed below is a Control-M/Agent user, then the <user> format is <username@HOST_ID>.

Viewing existing users in the Control-M Security database

This procedure describes how to view existing users in the Control-M Security database.

➢ To view existing users in the Control-M Security database:

1. Select Option 1 from the User Maintenance menu.

A list similar to the following is displayed:

```
Name        Description                    Group
GCSERV      For passing Global conds.      Group1
User1                                      Group2
Press ENTER to continue:
```

2. Press **<Enter>** to return to the User Maintenance menu.

The User Maintenance menu is redisplayed.

Adding a new user to the Control-M Security database

This procedure describes how to add a new user in the Control-M Security database.

➢ To add a new user to the Control-M Security database:

1. Select Option 2 from the User Maintenance menu.

A prompt similar to the following is displayed:

User []:

2. Specify the user name of the Control-M/EM user (maximum 64 characters, case-sensitive).

A prompt similar to the following is displayed:
User 'User2' is not defined in the Control-M Security database.
Add this user now [Y/N]?

3. Enter Y to add the new user.

The following prompts are displayed:

Description []:
Group []:

4. Specify a value for Description (maximum length: 50 characters) or press **<Enter>**.

   This field is optional and is for documentation purposes only.

5. Specify a value for Group (maximum length: 32 characters) or press **<Enter>**.

   This field is optional. If specified, the user inherits all authorizations defined for the group that are not specifically defined for the user.

6. Press **<Enter>** to return to the User Maintenance menu.

The User Maintenance menu is redisplayed.

Deleting an existing user from Control-M Security database

This procedure describes how to delete an existing user in the Control-M Security database.

➢ To delete an existing user from the Control-M Security database:

1. Select Option 3 from the User Maintenance menu.

   A prompt similar to the following is displayed:

   Username []:

2. Specify the user name of the Control-M/EM user to delete. After confirmation, the user is deleted from the Security database.

The User Maintenance menu is displayed.

Modifying the description of group fields for an existing user

This procedure describes how to modify the description or group fields in the Control-M Security database.

➢ To modify the Description or group fields for an existing user:

1. Select Option 4 from the User Maintenance menu.

   A prompt similar to the following is displayed:

   User []:

2. Specify the user name of the Control-M/EM user to modify.

   A user definition similar to the following is displayed:

```
User: User1
-----------------------

Modify User Information

1) Description :

2) Group       :

s) Save and return to menu

c) Cancel and return to menu
```

```
Enter command, or specify item number to modify:
```

3. Type the number preceding the field.

   You are prompted to supply a value for the field.

   - Maximum length for Description is 50 characters. The Description field is for documentation purposes only
   - Maximum length for Group is 32 characters.

4. Do one of the following:

   a. Type s to save your changes and return to the previous menu. Modifications are not saved until you perform this action.

   b. Type c to cancel all changes and return to the previous menu.

The previous menu is displayed.

Copying an existing user in Control-M Security database

This procedure describes how to copy an existing users in the Control-M Security database.

➢ To copy an existing user:

1. Select Option 5 from the User Maintenance menu.

   The following prompt is displayed:

   FROM user:

2. Specify the exact name of the user to be copied.

   The following prompt is displayed:

   TO user:

3. Specify a new user name for the Control-M/EM user (maximum 30 characters, case-sensitive).

   A prompt similar to the following is displayed:
   User 'User2' is not defined in the Control-M Security database.
   Add this user now [Y/N]?

4. Enter Y to add the new user. The following prompt is displayed:

   Description []:

5. Specify a value for Description (maximum length 50 characters). This field is optional and is for documentation purposes only.

   The following prompt is displayed:

   Group []:

6. Specify a value for Group (maximum length 32 characters).

   This field is optional. If specified, the user inherits all authorizations defined for the group that are not specifically defined for the user.

7. Press **<Enter>** to return to the User Maintenance menu.

Group maintenance

Each user who has a user account on the Control-M/Server computer and who is defined in the Control-M Security database, can be defined as part of a group. Belonging to a group is optional. All users belonging to a group inherit the authorizations defined for the group.

Select Option 2 from the Security Maintenance Main Menu to display the Group Maintenance menu.

```
Group Maintenance Menu
-----------------------
1)    List   Groups
2)    Add    Group
3)    Delete Group
4)    Modify Group Information

q)    Quit

Enter option:
```

Viewing existing groups in the Control-M Security database

This procedure describes how to view existing groups in the Control-M Security database.

➢ To view existing groups in the Control-M Security database:

1. Select Option 1 from the Group Maintenance menu.

   A list similar to the following is displayed:

   ```
   Group      Description
   Group1   Control-M Group
   Group2
   Press ENTER to continue:
   ```

2. Press <Enter> to return to the Group Maintenance menu.

The Group Maintenance menu is displayed.

Adding a new group to the Control-M Security database

This procedure describes how to add a new group in the Control-M Security database.

➢ To add a new group to the Control-M Security database:

1. Select Option 2 from the Group Maintenance menu. The following prompt is displayed:

   Groupname []:

2. Specify the Group name (maximum length 32 characters).

   This name must be unique. It cannot be an existing user or group name. The following prompt is displayed:

   Description []:

3. Specify a Description (maximum length 50 characters) or press <Enter>. The Description field is optional and is for documentation purposes only.

The group is added to the Security database, and the Group Maintenance menu is displayed.

Deleting an existing group from the Control-M Security database

This procedure describes how to delete an existing group in the Control-M Security database.

➢ To delete an existing group from the Control-M Security database:

1. Select Option 3 from the Group Maintenance menu. A prompt similar to the following is displayed:

   Group [Group1]:

2. Specify the name of the group to delete.

After confirmation, the group is deleted from the Security database, and the Group Maintenance menu is displayed.

Modifying the description field for an existing group

➢ To modify the Description field for an existing group:

1. Select Option 4 from the Group Maintenance menu. A prompt similar to the following is displayed:

   Group name []:

2. Specify the name of the group to modify.

   A group definition similar to the following is displayed:

   ```
   Group: Group1
   Modify Group Information
   -----------------------
   1) Description :
   s)  Save   and return to menu
   c)  Cancel and return to menu
   Enter command, or specify item number to modify:
   ```

3. Type 1

   You are prompted to supply a value for the field (maximum length 50 characters). This field is optional and is for documentation purposes only.

4. Do one of the following:

   a. Type s to save your changes and return to the previous menu. Modifications are not saved until you perform this action.

   b. Type c to cancel all changes and return to the previous menu.

Folder authorization

This option is used to assign authorizations to a user or group to perform actions on a Folder.

For more information about the types of authorization that can be granted using this option, see Folder Authorization options in Security maintenance utility (Batch mode) (on page 584).

Maintaining folder authorizations in Control-M Security database

This procedure describes how to maintain folder authorizations in the Control-M Security database.

➢ To maintain Folder authorizations:

1. Select Option 3 from the Main Menu. A prompt similar to the following is displayed:

```
+----------------------------------------+
|                 FOLDER AUTHORIZATION    |
+----------------------------------------+
```

   User/Group [User1]:

2. Specify the user or group for whom you are defining authorizations.

   If the user or group is not defined in the Control-M Security database, the following message is displayed:

   User/Group ‘name’ is not defined in the Control-M Security database.
   Press ENTER to continue:

3. Press <Enter> to return to the Main Menu. The Folder Authorization menu is displayed:

```
 Folder Authorization Menu
-----------------------------------
1)   List Folders
2)   Create/Modify Folder Authorization
3)   Delete Folder Authorization
q)   Quit
Enter option:
```

The relevant information is displayed according to the option you select.

Viewing existing Folder authorizations for the specified user/group:

This procedure describes how to view existing folder authorizations for the specified user/group in the Control-M Security database.

➢ To view an existing Folder authorizations for the specified user/group:

1. Select Option 1 from the Folder Authorization menu.

   A list similar to the following is displayed:

```
List for user: User1

Folder  Delete Read Update OrderFolder

Sched1    N      N      N         N

Sched2    N      Y      Y         Y

Press ENTER to continue:
```

2. Press **<Enter>** to return to the Folder Authorization menu.

Creating or modifying Folder authorizations for the specified user/group

This procedure describes how to create or modify folder authorizations for the specified user/group in the Control-M Security database.

➢ To create or modify Folder authorizations for the specified user/group:

1. Select Option 2 from the Folder Authorization menu. The following prompt is displayed:

   Folder Name:

2. Specify the name of a folder (maximum 20 characters, case-sensitive). The folder does not have to exist at the time you specify authorizations for it.

   A folder definition similar to the following is displayed:

```
 Folder Name: Sched1, User/Group: User1

Create/Modify Folder Authorization

---------------------------------

1)  Delete      :N

2)  Read        :N

3)  Update      :N

4)  OrderFolder  :N

s)  Save   and return to menu

c)  Cancel and return to menu

Enter command, or specify item number to toggle Y/N/D:
```

The Y setting enables authorization for the action (for example, Read), N disables the authorization, and (D)efault uses the authorization defined for the user's group. If the user was previously authorized for this folder, the user's current authorizations are displayed; otherwise, all authorizations are set to N.

To modify an authorization, type the number preceding the authorization and press **<Enter>**. (You may need to do this more than once in order to achieve the desired authorization.)

Type s to save your changes and return to the previous menu. Modifications are not saved until you perform this action.

-or-

Type c to cancel all changes and return to the previous menu.

Deleting Folder authorizations for the specified user/group

This procedure describes how to delete folder authorizations for the specified user/group in the Control-M Security database.

➢ To delete Folder authorizations for the specified user/group:

1. Select Option 3 from the Folder Authorization menu. The following prompt is displayed:

   Folder Name:

2. Specify the name of the Folder whose authorizations you want to delete for this user or group (or press **<Enter>** to return to the menu).

The user's authorizations for this folder are deleted from the Security database, and the Folder Authorizations menu is displayed. If the user belongs to a group, authorizations for the Folder revert to the authorizations defined for the group.

Active Jobs database authorization option

This option is used to assign authorizations to a user or group for actions on jobs in the Active Jobs database. The authorizations assigned are with regard to specific job owners (the user appearing in the Owner parameter for each job).

For more information about the types of authorization that can be granted using this option, see Active Jobs Authorization in Security maintenance utility (Batch mode) (on page 584).

Maintaining Active Jobs Database authorizations:

This procedure describes how to maintain Active Jobs Database authorizations.

➢ To maintain Active Jobs Database authorizations:

1. Select Option 4 from the Main Menu.

   A prompt similar to the following is displayed:

   ```
   +---------------------------------------+
   |    ACTIVE JOBS DATABASE AUTHORIZATION    |
   +---------------------------------------+
   ```

   User/Group[]:

2. Specify the user or group for whom you are defining authorizations.

   If the user or group is not defined in the Control-M Security database, the following message is displayed:
   User/Group 'name' is not defined in the Control-M Security database.
   Press ENTER to continue:

3. Press **<Enter>** to return to the Main Menu. The Active Jobs database Authorization menu is displayed:

   ```
   Active Jobs Database Authorization Menu
   -----------------------------------
   1)  List Owner Names
   2)  Create/Modify AJF Authorization
   3)  Delete AJF Authorization
   q)  Quit

   Enter option:
   ```

Viewing owners for whom the user has Active Jobs Database authorizations

This procedure describes how to view owners for whom the user has Active Jobs Database authorizations.

➢ To view owners for whom the user has Active Jobs Database authorizations:

- Select Option 1 from the Active Jobs database Authorization menu.

  A list similar to the following is displayed:

```
List for user: User1
Owner Host Group Hold Force Del Rerun Log Why Statist Output Order Conf Z&S
Kill
----- ---------- ---- ----- --- ----- --- --- ------- ------ ----- ---- ---
----
Owner1  Host1      N     Y    Y     Y    N   N     N        N     Y     N
N   Y
Owner2  Host2      Y     Y    Y     Y    Y   Y     Y        Y     Y     Y
N   N
Press ENTER to continue:
```

The modify Active Jobs Database authorizations is displayed.

Creating or modifying Active Jobs Database authorizations for the specified user

This procedure describes how to create or modify Active Jobs Database authorizations for the specified user.

➢ To create or modify Active Jobs Database authorizations for the specified user:

1. Select Option 2 from the Active Jobs database Authorization menu.The following prompt is displayed:

   Owner:

2. Specify the name of a job owner.

   The following prompt is displayed:

   Host Group:

3. Specify the host group of the Agents where the job can be scheduled to run (maximum 30 characters, case-sensitive).

   A value must be specified for the Host Group prompt. To indicate all host groups, specify an asterisk (*) for this prompt.

   A list similar to the following is displayed:

```
Owner: Owner1, Host Group: Host1 User/Group:

1)  Order          :Y
2)  Force          :Y
3)  Rerun          :Y
4)  Hold           :N
5)  Confirm        :N
6)  Delete         :Y
7)  Why            :N
8)  Output         :N
9)  Log            :N
10) Statistics     :N
11) Zoom & Save    :N
12) Kill job       :N

s)  Save   and return to menu

c)  Cancel and return to menu

Enter command, or specify item number to toggle Y/N/D:
```

The Y setting enables authorization for the action (for example, Read), N disables the authorization, and (D)efault uses the authorization defined for the user's group. If the user was previous authorized for this owner and host, the user's current authorizations are displayed; otherwise, all authorizations are set to N.

When working in full security mode and ordering SMART Folders where **Y** has been specified for **Order**, BMC recommends to specify **Y** also for **Hold**. The SMART Folder remains in **Hold** status if the user has only ORDER/FORCE permissions. In addition, if you did not specify an asterisk (*) for the **Host Group** prompt, you need to create another Active Jobs database authorizations for the specified user for the SMART folder and Sub-folder entities, in which in the **Host Group** prompt, you must specify the local hostname of the Control-M/Server. Do this by running ctm_menu and then from the Control-M Main Menu select option **5 - Parameter Customization**, then option **1 - Basic Communication and Operational Parameters** and then **1 - Local IP Host Interface Name**.

4. To modify an authorization, type the number preceding the authorization and press **<Enter>** (you may need to do this more than once in order to achieve the desired authorization).
5. Do one of the following:
   - Type s to save your changes and return to the previous menu.
   - Type c to cancel all changes and return to the previous menu.

Deleting Active Jobs Database authorizations for the specified user

This procedure describes how to delete Active Jobs Database authorizations for the specified user.

➢ To delete Active Jobs Database authorizations for the specified user:

1. Select Option 3 from the Active Jobs database Authorization menu.

   The following prompt is displayed:

   Owner:

2. Specify the name of the Job owner for whom authorizations should be deleted (or press <Enter> to return to the menu).

   The following prompt is displayed

   Host Group:

3. Specify the name of the Host group of the Job owner for whom authorizations should be deleted (or press <Enter> to return to the menu).

   The user authorizations for this owner on the Host group are deleted from the Security database, and the Active Jobs database Authorization menu is displayed.

Entities authorization

This option assigns authorizations to a user or group to perform actions relating to Control-M entities.

For more information about the types of authorization that can be granted using this option, see Entities Authorization Option in Security maintenance utility (Batch mode) (on page 584).

Maintaining Entities authorizations

This procedure describes how to maintain Entities authorizations.

➢ To maintain Entities authorizations:

1. Select Option 5 from the Main Menu.

   A prompt similar to the following is displayed:

   ```
   +-----------------------------------------+
   |    Control-M ENTITIES AUTHORIZATION     |
   +-----------------------------------------+
   User/Group [User1]:
   ```

2. Specify the user or group for whom you are defining authorizations.

   - If the user or group name is not defined on the server computer, a message similar to the following is displayed:

     User 'name' is not defined in the Control-M Security database
     Press ENTER to continue:

     Press **<Enter>** to return to the Main Menu.

   - If the user or group name is defined the Entities Authorizations menu is displayed:

     ```
     Entities Authorizations Menu
     ----------------------------
     1)  List Entity Categories
     2)  Create/Modify Entity Authorizations
     3)  Delete Entity Category
     q)  Quit
     Enter option:
     ```

The relevant information is displayed according to the option you select.

Viewing Entity categories for which the user or group has authorizations

This procedure describes how to view categories for which the user or group has authorizations.

➢ To view Entity categories for which the user or group has authorizations:

1. Select Option 1 from the Entities Authorizations menu.

   A list similar to the following is displayed:

   ```
   List for user: User1

   Category               Add Delete Change

   CALENDAR                Y    N       Y

   QUANTITATIVE RESOURCE Y    N       N

   Control RESOURCE        Y    N       N

   Press ENTER to continue:
   ```

2. Press **<Enter>** to return to the Entities Authorizations menu.

Creating or modifying Entity authorizations for the specified user or group

This procedure describes how to create or modify Entity authorizations for the specified user or group.

➢ To create or modify Entity authorizations for the specified user or group:

1. Select Option 2 from the Entities Authorizations menu. The following menu is displayed:

```
Categories
----------
1) CALENDAR
2) LOG
3) QUANTITATIVE RESOURCE
4) CONDITION
5) Control RESOURCE
q) Quit
Category number:
```

2. Specify the number of the category for which to create or modify authorizations. For example, if you specify 1, a list similar to the following is displayed:

```
Category: CALENDAR, User: User1
Create/Change Entity Authorizations
-----------------------------------
1)  Add    :Y
2)  Delete :N
3)  Change :N
s)  Save   and return to menu
c)  Cancel and return to menu
Enter command, or specify item number to toggle Y/N/D:
```

   The Y setting enables the specific authorization (for example, Read), N disables the authorization, and (D)efault uses the authorization defined for the group with which the user is associated. If the user was previous authorized for this category, the user’s current authorizations are displayed; otherwise, all authorizations are set to N.

3. To modify an authorization, type the number preceding the authorization and press **<Enter>** (you may need to do this more than once in order to achieve the desired authorization).

4. Do one of the following:

   - Type s to save your changes and return to the previous menu. Modifications are not saved until you perform this action.
   - Type c to cancel all changes and return to the previous menu.

Deleting Entity authorizations for the specified user or group

This procedure describes how to delete Entity authorizations for the specified user or group.

➢ To delete Entity authorizations for the specified user or group :

1. Select Option 3 from the Entities Authorizations menu. The following menu is displayed:

   ```
   Categories
   ----------
   1) CALENDAR
   2) LOG
   3) QUANTITATIVE RESOURCE
   4) CONDITION
   5) Control RESOURCE
   q) Quit
   Category number:
   ```

2. Specify the number of the category for which to delete authorizations and press **<Enter>**.

   The user or group’s authorizations for this category are deleted from the Security database and the Entities Authorizations menu is displayed.

Security maintenance utility (Batch mode)

Certain ctmsec Security Maintenance utility functions can be activated in batch mode. These functions include listing, updating, and deleting entries in the Control-M Security database. These functions are described inSecurity maintenance utility (Interactive mode) (on page 570) .

User authorization

The user authorization options of the ctmsec command are used to list, update, delete, and copy users in the Control-M Security database.

- Use the following command to list user authorizations:
  ```
  ctmsec -USER_LIST <user>
  ```
- Use the following command to update user authorizations:
  ```
  ctmsec -USER_UPDATE <user> <description> <group>
  ```
- Use the following command to delete user authorizations:
  ```
  ctmsec -USER_DELETE <user>
  ```
- Use the following command to copy user authorizations from one user to another:
  ```
  ctmsec -USER_COPY <from_user> <to_user>
  ```

If the user in the commands listed above is a Control-M/Agent user, then the <user> format is <username@host_id>.

Group authorization

Group authorization options of the ctmsec command are used to list, modify, and delete groups in the Control-M Security database.

- Use the following command to list group authorizations:
  ```
  ctmsec -GROUP_LIST <group>
  ```
- Use the following command to update group authorizations:
  ```
  ctmsec -GROUP_UPDATE <group> <description>
  ```
- Use the following command to delete group authorizations:
  ```
  ctmsec  -GROUP_DELETE <group>
  ```

Folder authorization option

The Folder authorization options of the ctmsec command are used to assign authorizations to users and groups to perform actions on Folders.

- Use the following command to list Folder authorizations:
  ```
  ctmsec -SCHED_LIST {<user>|<group>}
  ```
- Use the following command to update Folder authorizations:
  ```
  ctmsec -SCHED_UPDATE {<user>|<group>}     <folder>     [-DELETE {Y|N|D}]
  [ -READ {Y|N|D}]
      [ -ORDER {Y|N|D}] [ -UPDATE {Y|N|D}]
  ```
- Use the following command to deleteFolder authorizations:
  ```
  ctmsec -SCHED_DELETE {<user>|<group>}     <folder>
  ```

If the user in the commands listed above is a Control-M/Agent user, then the <user> format is <username@host_id>.

Active Jobs database authorization

The Active Jobs database authorization options of the ctmsec command are used to assign authorizations to users and groups to perform actions on jobs in the Active Jobs database.

- Use the following command to list Active Jobs database authorizations:
  ```
  ctmsec -ACT_LIST {<user>|<group>}
  ```
- Use the following command to update Active Jobs database authorizations:
  ```
   ctmsec -ACT_UPDATE {<user>|<group>} <owner> <host>
   [-HOLD          {Y|N|D}]
    [-FORCE          {Y|N|D}]
   [-ORDER         {Y|N|D}]
       [-CONFIRM        {Y|N|D}]
  [-DELETE        {Y|N|D}]
   [-WHY           {Y|N|D}]
   [-RERUN         {Y|N|D}]
   [-OUTPUT        {Y|N|D}]
   [-LOG           {Y|N|D}]
   [-STATISTICS    {Y|N|D}]
  [-ZOOM_AND_SAVE {Y|N|D}]
   [-KILL_JOB      {Y|N|D}]
  ```
- Use the following command to delete Active Jobs database authorizations:
  ```
  ctmsec -ACT_DELETE {<user>|<group>} <owner> <host>
  ```

If the user in the commands listed above is a Control-M/Agent user, then the <user> format is <username@host_id>.

Entities authorization options

The entity authorization options of the ctmsec command are used to assign authorizations to users and groups to perform actions relating to Control-M entities.

- Use the following command to list entity authorizations:
  ```
  ctmsec -ENTITY_LIST {<user>|<group>}
  ```
- Use the following command to update entity authorizations:
  ```
  ctmsec -ENTITY_UPDATE     {<user>|<group>}
      {LOG|QR|Control|CALENDAR|CONDITION}
   [-ADD {Y|N|D}] [-DELETE {Y|N|D}] [-CHANGE {Y|N|D}]
  ```
- Use the following command to delete entity authorizations:
  ```
  ctmsec -ENTITY_DELETE    {<user>|<group>}
      {LOG|QR|Control|CALENDAR|CONDITION}
  ```

If the user in the commands listed above is a Control-M/Agent user, then the <user> format is <username@host_id>.

Exporting security definition folders

The EXPORT option of the ctmsec command is used to export Control-M Security Definition folders. The file that is generated by the ctmsec command is an execufolder file containing API functions that will redefine all the security entries when the script is run. The generated file can be modified and imported to any Control-M installation.

The file created by the EXPORT option of the ctmsec utility can be modified before security definitions are imported back to the same (or a different) Control-M/Server installation. This is different from the file that is created using the Backup Security Definition Folders option of the Security Authorization Menu (which cannot be modified).

Use the following command to export Control-M Security Definition folders:

```
ctmsec -EXPORT <fileName>
```

`<fileName>` is the full path name of the file to be exported.

```
ctmsec -EXPORT /home/controlm/securedata
```

Importing security definition folders

The file created using the -EXPORT option of the ctmsec utility contains multiple ctmsec commands that describe the various security definitions in your Control-M installation. If necessary, these ctmsec commands can be modified before the security definitions are imported back to the same or a different Control-M installation.

Importing updates the security definitions in your Control-M installation. Use the restore security procedure to replace security definitions.

➢ To import security definitions:

- Execute the script file that was created using the ctmsec utility.

```
/home/controlm/securedata
```

This procedure will work only with a file that was created using the -EXPORT option of the ctmsec utility. If your input is a file created using the Backup Security Definition Folders option of the Security Authorization menu, then you must import using the Restore option in that same menu.

ctmsetown

The ctmsetown command line utility manages the authentication credentials of job owners for both local and agentless jobs. In addition, the ctmsetown utility also enables the authentication details of users to be imported or exported from different Control-M environments. To run the ctmsetown utility see: Running the ctmsetown utility (on page 587).

If the ctmsetown utility is invoked by a job, the passwords should be defined in an encrypted form to avoid exposing them when the job submission is sent over the network.

When a job is submitted, Control-M/Server attempts to find the owner and hostname authentication details.

- If the owner and hostname are found, Control-M/Server uses these credentials.
- If the specified hostname is not found, Control-M/Server tries to find the owner on host <All>.
- If the owner is found on host <All>, Control-M/Server uses these credentials.
- If the owner is not found on the specified hostname or on host <All>, Control-M/Server uses empty credentials.

Using ctmsetown through Control-M/Agent, the functionality of the ctmsetown utility (when invoked from Control-M/Agent), is limited to updating passwords of existing owners. Using the utility, job owner passwords can be updated through Control-M/Agent for:

- Jobs running on agentless hosts
- Jobs running on Control-M/Agent for Windows that is configured to work in 'logon as user' mode"
- Jobs running on Control-M/Agent for UNIX that is running in non-root mode

Running the ctmsetown utility

This procedure descrbies how to run the ctmsetown utility, which manages the authentication credentials of job owners for both local and agentless jobs.

➢ To run the ctmsetown utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Type one of the following commands when working with Control-M/Server:
   - o
     ```
     ctmsetown -action add -owner <userName> -host <hostName>
     [-password <password> | -encpassword <encryptedPassword> | -keyname
     <keyName> [-passphrase <keyPassphrase> | -encpassphrase
     <encryptedKeyPassphrase>]]
     ```
   - o
     ```
     ctmsetown -action update -owner <userName> -host <hostName>
     [-password <password> | -encpassword <encryptedPassword> | -keyname
     <keyName > [-passphrase <keyPassphrase> | -encpassphrase
     <encryptedKeyPassphrase>]]
     ```
   - o
     ```
     ctmsetown -action delete -owner <userName> -host <hostName>
     ```

- `ctmsetown -action list [-owner <userName>] [-host <hostName>]`
- `ctmsetown -action export -filename <exportFileName>`
- `ctmsetown -action import -filename <importFileName> -data append|truncate`
- `ctmsetown help`

3. Specify the following command to invoke the ctmsetown utility from Control-M/Agent:

```
ctmsetown -action update -owner <user name>
-host <host name> -password <new password>
-oldpassword <old password>
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on ctmsetown utility, see: ctmsetown utility action parameters (on page 588), ctmsetown utility parameters (on page 589) and ctmsetown utility examples (on page 590).

ctmsetown utility action parameters

The following table describes the actions in the ctmsetown utility:

Action	Description
add	Specifies the security details of a new owner entry (user).
update	Modifies the security details of an existing owner entry (user).
delete	Removes the security details of an owner entry. The owner name and host name must match an existing entry in the folder.
list	Lists the details of the user. Wildcards can be used to specify -owner and -host parameters, as follows: ▪ * represents any number of characters ▪ ? represents any single character
export	Exports the security details of the existing users to a text file. `ctmsetown -action export -filename $HOME/ctm_server/data/user_report.txt`
import	Imports the details of the users stored in the specified import file.
help	Displays the usage of the ctmsetown utility.

ctmsetown utility parameters

The following table describes the ctmsetown utility parameters:

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td>-run as</td><td colspan="2">Specifies the name of the user under whose name the job will run.</td></tr>
<tr><td>-host</td><td colspan="2">Specifies the name of the computer where the owner of the job is defined. Specify <All> to include all hosts.
ctmsetown -action delete -owner s -host "<All>"</td></tr>
<tr><td>-password</td><td colspan="2">Specifies the password of the owner. The password cannot exceed 120 characters.</td></tr>
<tr><td>-oldpassword</td><td colspan="2">Specifies the existing password that the user is changing. This parameter is mandatory only when the ctmsetown utility is executed from the agent.</td></tr>
<tr><td>-encpassword</td><td colspan="2">The encrypted password of the owner. The encpassword cannot exceed 512 characters. The value must be divisible by 4 without a remainder.</td></tr>
<tr><td rowspan="3">-keyname</td><td colspan="2">The logical name of the key. The key itself is kept in a separate folder with its passphrase. For more information about generating and maintaining the key, see ctmkeygen (on page 426). The same key can be used for multiple users.
If -keyname is specified, then specify one of the following parameters:</td></tr>
<tr><td>-passphrase</td><td>Phrase used as a key to encrypt the key itself.</td></tr>
<tr><td>-encpassphrase</td><td>The encrypted phrase used as a key to encrypt the key itself. The encpassphrase cannot exceed 512 characters. The value must be divisible by 4 without a remainder.</td></tr>
<tr><td>-filename</td><td colspan="2">Specifies the name of the file that contains the security details of the users. The filename cannot exceed 1024 characters.
This parameter is used only when either -action export or -action import is specified.</td></tr>
<tr><td rowspan="3">-data</td><td colspan="2">Describes what action to take with the data from the imported text file. Valid actions are:</td></tr>
<tr><td>append</td><td>details of the users from the imported text file are added to the existing users</td></tr>
<tr><td>truncate</td><td>details of the users from the imported text file replace the details of the existing users</td></tr>
</table>

ctmsetown utility examples

The following are examples of the ctmsetown utility commands that are run from Control-M/Server, apart from the last example which is run from Control-M/Agent.

To create an entry with the security details of a user whose name is username1, the name of the host computer is saturn and the user password is pass01, specify the following command:

```
ctmsetown -action add -owner username1 -host saturn -password pass01
```

The following message is displayed:

```
Entry created successfully.
```

Create a user entry as in the first example, however, use the keyname k1 and passphrase BMC user. Specify the following command:

```
ctmsetown -action add -owner username1 -host saturn -keyname k1
-passphrase "BMC user"
```

The following message is displayed:

```
Entry created successfully.
```

Assume that the security details of the owner, described in the first example, already exists. To change the password from pass01 to newpass, specify the following command:

```
ctmsetown -action update -owner username1 -host saturn -password
newpass
```

The following message is displayed:

```
Entry updated successfully.
```

To delete the user entry created in the first example, specify the following command:

```
ctmsetown -action delete -owner username1 -host saturn
```

The following message is displayed:

```
Entry deleted successfully.
```

To list the security details of user entries, specify the following command:

```
ctmsetown -action list

Owner                 Host                  Password/Key Flag
Key value

-----                 ----                  -----------------
---------

jupiter               saturn                Key
Key1

jupiter               venus                 Password
Not Applicable

2 entries were found.
```

To create an export text file containing a list of security details of user entries, specify the following command:

```
ctmsetown -action export -filename /home/ctm630oe/sec.exp
```

The following is displayed:

```
Exporting data, please wait...

Export ended successfully.

Check report file
~<controlm_owner>/ctm_server/proclog/export_report_53d1.txt' for
details.
```

To import the **/home/ctm630oe/sec.exp** text file created in the sixth example, containing a list of security user entries, and to replace the current security user information, specify the following command:

```
ctmsetown -action import -filename /home/ctm630oe/sec.exp -data
truncate
```

The following is displayed:

```
Importing data, please wait...

Import ended successfully.

Check report file
~<controlm_owner>/ctm_server/proclog/import_report_53d9.txt' for
details.
```

Example to show ctmsetown run from an agent computer to update the password of a user.

Assume that the old password of user agentuser1 is agntpass01. To change the password to newpass, specify the following command:

```
ctmsetown -action update -owner agentuser1 -host saturn -password
newpass
```

The following message is displayed:

```
Entry updated successfully.
```

Control-M/Agent utilties

Control-M/Agent utility for security:

- ctmpwd (on page 591)

ctmpwd

The ctmpwd utility (Windows only) adds, updates, and deletes Control-M/Agent users and passwords. In addition, it changes security settings for the agent directories and cmd.exe. It also lists all users in the Control-M/Agent password file. (This utility replaces the ctmcpt utility in earlier versions.)

To run ctmpwd, you must be an administrator on the computer. In addition, Windows 2000 users user running the utility must have the proper privileges defined in the Act as part of operating system parameter of the Local Security Settings application on the target computer.

Running the ctmpwd utility

This procedure describes how to run the ctmpwd utility, which enabes you to

Before you begin

- You must manually give **Logon as a batch job** rights to a new user.

➢ To run the ctmpwd utility:

- Run the following commands:

  -
    ```
    CTMPWD -ACTION ADD|UPDATE|DELETE|LIST [-USER <user name>]
    [[-OLD_PASSWORD <value>] -PASSWORD <value>] [-ADMIN_PASSWORD <value>]]
    -AGENT <agent name> -GROUP <group name>
    ```
  - Add a user and password

    ```
    ctmpwd -action add -user user1 -password 12345
    ```
  - Add the administrator user

    ```
    ctmpwd -action add -user admin -password abcde
    ```
  - Update a password

    ```
    ctmpwd -action update -user user1 -old_password 12345 -password 67890
    ```

    or

    ```
    ctmpwd -action update -user user1 -admin_password abcde -password 67890
    ```
  - Delete a user

    ```
    ctmpwd -action delete -user user1 -password 12345
    ```

    or

    ```
    ctmpwd -action delete -user user1 -admin_password abcde
    ```
  - List all users

    ```
    ctmpwd -action list
    ```
  - Add a user to agent Saturn

    ```
    ctmpwd -action add -user user3 -password 654321 -agent Saturn
    ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on ctmpwd utility, see ctmpwd parameters (on page 593) and ctmpwd examples (on page 593).

ctmpwd parameters

The following table describes the ctmpwd utility parameters:

Parameter	Description
action	Function to be executed. Valid values: add, update, delete, and list.
user	Name of the user. When adding users, the user name must not exceed 20 characters.
old_password	Current password for the update function.
password	Current password for the delete function. New password for the add and update functions.
admin_password	Password for the Control-M/Agent administrator when executing the update or delete function if the old_password is not known.
verify	Verifies that the user and password. Valid values are: ▪ Y (default) ▪ N - Does not verify that the user and password are correct.
sub_application	Adds the group SID (instead of the user SID) to cmd.exe and the agent directories.
agent	Name of the agent that the utility is designated to run on. For more information, see Considerations for running utilities (on page 18).

ctmpwd examples

In the following example, the ctmpwd utility enables the Control-M/Agent administrator to modify passwords for users who have forgotten their password.

-admin_password

BMC recommends that the administrator first use the following command to establish a password for user ADMIN:

```
ctmpwd -action add -user ADMIN -password <user_admin_password>
```

In the following example, the user is added but the group's SID is registered.

```
ctmpwd –action add –user user1 –password user1 –group Everyone
```

Statistics and reporting

The statistics and reporting utilities generate and display various statistics.

By including a utility command in the command line of a job processing definition, you can run the utility at a predetermined time or under a predetermined set of conditions without being present.

Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Utilities that generate and display various statistics

Utility	Description
bim_report (on page 594)	Generates reports for services that have completed execution.
ctmjsa utility (on page 602)	Compiles and records runtime data from the Statistical Details table.
ctmruninf (on page 605)	Displays runtime data from the Statistical Details table of the Control-M/Server database.
ctmstats (on page 608)	Displays and deletes statistical data from the Statistical Summary table of the Control-M/Server database.
emreportcli (on page 597)	Enables for the automation of the batch report execution process in a selected format.

bim_report

Generates reports for services that have completed execution.

The bim_report utility can be run both on Microsoft Windows and UNIX operating systems from their respective command lines. To run the utility see: Running the bim_report utility (on page 594).

Running the bim_report utility

This procedure describes how to run the bim report utility, which enables you to generates reports for services that have completed execution.

➢ To run the bim_report utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the following command:

```
bim_report -U username
-P password
[-O output file name]
[-N service name]
[-F from date]
```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the bim_report utility, see bim_report utility parameters (on page 596) and bim_report utility example (on page 596).

bim_report utility parameters

The following table describes the bim_report utility parameters:

Parameter	Description
-U *username*	Name of the Control-M/EM user running the report, for security purposes. Mandatory.
-P *password*	Password of the Control-M/EM user running the report, for security purposes. Mandatory.
-O *output file name*	Name and full path of the output file that will contain the generated report. Optional. If not specified, the report is displayed on the screen but is not saved in a file.
-N *service name*	Name of the service that should be listed in the report. Optional. If not specified, the report runs for all services.
-F *from date*	Services whose order date (Odate) is the same as, or later than, this from date value are included in the report. The date and time format is DD/MM/YYYY_HH:MI:SS or MM/DD/YYYY_HH:MI:SS format (depending on the value of the DateFormat system parameter). Optional. If not specified, services whose order date was on the previous day or later are included in the report.
-T *to date*	Services whose order date (Odate) is the same as, or earlier than, this to date value are included in the report. The date and time format is DD/MM/YYYY_HH:MI:SS or MM/DD/YYYY_HH:MI:SS format (depending on the value of the DateFormat system parameter). Optional. If not specified, services whose order date is the current date and time or earlier are included in the report.

bim_report utility example

The following example describes running the bim_report utility parameters in Windows:

```
bim_report
-U emuser
-P empass
-O D:\Temp\my_report.txt
-N CD_service
```

```
-F 26/02/2004_08:34:00
-T 28/02/2004_23:34:00
```

emreportcli

The emreportcli utility enables the automation of the batch report execution process in a selected format. This utility runs only on Windows in batch mode. To run the emreportcli utility, see: Running the emreportcli utility (on page 597).

You must specify Control-M/EM server login information when you invoke this utility.

Usage: Either of the following two options can be used to activate the emreportcli utility:

- Parameters can be entered as command line parameters
- An input arguments file can be generated with XML specifications. For more information, see: emreportcli input arguments file (on page 599)

Running the emreportcli utility

This procedure describes how to run the emportcli utility, which enables the automation of the batch report execution process in a selected format.

➢ To run the emreportcli utility:

1. Open a command prompt window where Control-M/EM is installed.
2. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
3. Enter one of the following command and press Enter.

   - `emreportcli [{(-U emUser -P emPass) | -pf passwordFilename}]`

   ```
   -s emServer -arg xmlArgumentsFile
   ```

   -
     ```
     emreportcli {-u <user> -p <password> | -pf <password file>} -s <server
     host name>
     -template  <templateName>
     [-template_path <template path>]
     -output_file_type EXCEL | EXCEL_DO | DOC | PDF | HTML |
     XML | CSV | TABBED
     -output_file_path <output file path>
     [-param <name>=<value>]…
     ```

   The xmlArgumentsFile is the full path name of the input arguments file.

You can specify the user name and password on the command line, in a password file. For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the emreportcli utility see the following:

- emreportcli report generation utility parameters (on page 598)
- emreportcli utility output argument file parameters (on page 598)
- emreportcli utility input arguments file parameters (on page 600)

emreportcli report generation utility parameters

The following table describes the emreportcli report generation utility parameters

Parameter	Description
emUser	Control-M/Enterprise Manager user name.
emPass	Control-M/Enterprise Manager user password.
PasswordFilename	Flat file containing an unencrypted username and password in the following format: user=username password=password. If both -U and -pf are specified, an error message is generated. If neither is specified, an online prompt is issued for the Control-M/EM database owner name and password.
emServer	Host name of the Control-M/EM Server. To address a GUI Server when multiple GUI Servers exist, set this parameter to the logical name of the relevant GUI Server.

emreportcli utility output argument file parameters

The following table describes the emreportcli utility output argument file parameters:

Parameter	Attribute	Description
OutputFile		The report output file.
	type	Specifies the type of the output file, such as EXCEL, EXCEL_DO (for data only), PDF, DOC, HTML, TXT, or XML.
	file path	Specifies the full filename of the output file.

Parameter	Attribute	Description
Param		Specifies the name and value for each parameter in the form name=value. Wildcard can be used for text fields.
Template		Specifies the name of the template.
	path	Specifies the folder in which the template file is located.

emreportcli input arguments file

Input for the utility is an arguments file containing the XML specifications for the report to be generated, including all required parameters. The following sample XML arguments file is provided at em_home\Data\Reporting\sample_args.xml. A list of required elements and values for the arguments is provided in the DTD file: em_home\Data\Reporting\emreportcli.dtd.

```
emreportcli {-u <user> -p <password> | -pf <password file>} -s
<server name> -arg <XML file name>
```

If you are using only a client installation you must include the ‘bin’ directory in the file path. For example:

c:\Program Files\<Instance Name>\bin\emreportcli.exe {-u <user> -p <password> | -pf <password file>} -s <GUI Server Name> -arg <arguments file>.xml}

```
<!DOCTYPE ReportDefinitions SYSTEM "emreportcli.dtd">
<ReportDefinitions>
       <ReportDefinition>
              <SourceFile templateName="alerts3"/>
              <OutputFile type="PDF" filepath="D:\MyAlerts3.pdf"/>
              <Parameters>
                    <Parameter name="MY_PARAM" value="job*"/>
                    <Parameter name="Application" value="a*"/>
              </Parameters>
       </ReportDefinition>
</ReportDefinitions>
```

With the input arguments file in the above example, the emreportcli generates a PDF report based on the "alerts3" report template and saves it in the D:\ folder with the filename called MyAlerts3.pdf.

If the report template filter is defined as shown in ZBlueXrefparanum, the output of the report will contain all alerts with a

- Control-M name of "ctm640" or that starts with "mvs"
- Job Name that starts with "job"
- Application name that starts with "a"

emreportcli utility input arguments file parameters

The following table describes the emreportcli utility input arguments file parameters:

Parameter	Attribute	Description
SourceFile		The template used to generate the report
	templateName	Specifies the name of the template
	TemplatePath	Specifies the folder where the template file is located (Optional)
OutputFile		The report output file.
	type	Specifies the type of the output file, such as EXCEL, EXCEL_DO (for data only), PDF, DOC, HTML, TXT, or XML
	filepath	Specifies the full filename of the output file (which will be overwritten if it previously existed) Dynamically resolved keys can be included when specifying the filepath attribute: {date}, {time}, and {counter}. For filepath=D:\Test.doc (that is, no key), the output file is: D:\Test.doc. For filepath=D:\Test-{date}.doc, the output file is D:\Test-May22, 2008.doc. For filepath=D:\Test{counter}.doc, the first output file is D:\Test1.doc, and the next output file is D:\Test2.doc.
Parameters		The parameter list.
Parameter		An individual parameter, whose name and value is specified.
	name	name of the parameter as defined in the report template filter panel.

Parameter	Attribute	Description
	value	value of the report parameter (wildcard characters can be used for text fields when the field operator in the filter panel is set to "LIKE" for the fields).

Control-M/Server utilities

This table lists the Control-M/Server utilities for statistics and reporting.

Utility Type	Description
ctmjsa utility (on page 602)	The ctmjsa utility compiles runtime data from the Statistical Details table and records it in the Statistics Summary table of the Control-M/Server database.
ctmruninf (on page 605)	The ctmruninf utility displays runtime data from the Statistical Details table of the Control-M/Server database.
ctmstats (on page 608)	The ctmstats utility displays and deletes statistical data from the Statistical Summary table of the Control-M/Server database.

ctmjsa utility

The ctmjsa utility compiles runtime data from the Statistical Details table and records it in the Statistics Summary table of the Control-M/Server database. This utility must be run by a Control-M/Server user.

Each time it is run, this utility:

- Scans the statistical data for jobs that terminated with OK status. The jobs scanned can be limited to a range of dates as described below.
- Computes the average run time and standard deviation for each job for which data was found.
- Records the statistical data in a summary table in the Control-M/Server database (from which the data is made available to Control-M/EM).
- No other output is generated by this utility.
- Display the summary data filtered according to specified parameters.

Statistical data is only accumulated when the Control-M system parameter Statistics is set to Y. Operational parameter Statistics Mode determines the mode to be used to compile summary statistics: JOBNAME or MEMNAME. The default is MEMNAME.

If the Statistics Mode parameter was changed from MEMNAME to JOBNAME or back since the last run of the ctmjsa utility, you can cleanup the statistics from the previous mode by running the following command: ctmstats –delete. The Statistics Mode parameter can be changed through ctm_menu by choosing **Parameter Customization Menu =>Advanced Communication and Operational Parameters =>Statistics Mode**.

For more information about runtime statistical data, see the information about runtime statistics in Control-M Forecast parameters. To run the ctmjsa utility, see: Running the ctmjsa utility (on page 602).

Running the ctmjsa utility

This procedure describes how to run the ctmjsa utility, which compiles runtime data from the Statistical Details table and records it in the Statistics Summary table of the Control-M/Server database.

➢ To run the ctmjsa utility:

1. Do one of the following:

a. Log on to a Control-M for Databases account (*UNIX*)

a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter the one of the following commands:

- `ctmjsa <fromDate> <toDate>`
- `ctmjsa -<delta1> -<delta2> <date>`
-
```
ctmjsa -list [ -MEMNAME <memname> ]
             [ -MEMLIB  <memlib> ]
             [ -HOSTID  <hostid> ]
```

If the Statistics Mode parameter is JOBNAME, Mem Name and Mem Lib, fields in the Statistical Summary table are blank. If the Statistics Mode parameter is MEMNAME, the Job Name field is blank.

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information, see: ctmjsa utility parameters (on page 603) and ctmjsa utility example (on page 604).

ctmjsa utility parameters

This table describes the ctmjsa utility parameters:

Parameter	**Description**	
<fromDate>	Starting date of statistical data to be compiled. The date is specified in yyyymmdd or yymmdd format.	
<toDate>	Ending date of statistical data to be compiled. The date is specified in yyyymmdd or yymmdd format.	
-*<delta1>*	Unsigned number used to establish the starting date for statistical data to be compiled. This date is determined by subtracting *<delta1>* from *<date>* (for example, if *<delta1>* is 10 and *<date>* is 991220, the starting date is 991210).	
-*<delta2>*	Unsigned number used to establish the ending date for statistical data to be compiled. This date is determined by adding <Delta2> to <Date> (for example, if <Delta2> is 5 and <Date> is 991220, the ending date is 991225).	
<date>	Date used together with <Delta1> and <Delta2> to determine the range of dates used for compiling statistical data. The date is expressed in yyyymmdd or yymmdd format.	
"*"	Asterisk enclosed in quotation marks. Specifies that the utility collects all statistical data available without regard to date.	
-list	Display data from the Statistical Summary table filtered according to specified subparameters. Use this option after you have updated the summary table. Output includes the Folder name for each job. This information is also available from Control-M/EM in the Statistics window.	
Filter	Specify one of the following options and its subparameter, or specify the null character " " to display statistics for all jobs. This works the same way as "*", which should be enclosed in quotation marks. ? Represents any single character.	
	-MEMNAME <memname>	Identify the job by its Mem Name parameter.
	-MEMLIB <memlib>	Identify jobs by their Mem Lib parameter.

Parameter	Description	
	-HOSTID <hostid>	Identify jobs by their host group parameter (agent computer).

ctmjsa utility example

The following example describes commands compile statistical data for the 5-day period from June 21, 2000 through June 25, 2000 (assuming this data is available). In the second command, the hyphens indicate the beginning of unsigned parameter values; they are not minus signs.

```
ctmjsa 000621 000625
```

```
ctmjsa -3 -1 000624
```

The following command compiles statistical data using all data currently available:

```
ctmjsa "*"
```

This command displays summary data for all jobs whose Mem Name parameter starts with "pgmac": ctmjsa -list -MEMNAME "pgmac*"
A report similar to the following is displayed:

JOBNAME	MEMNAME	MEMLIB	HOSTID	CPU [sec]	ELAPSED (sec)	FOLDER
	pgmacct1	prod.acct.pgm	diana	0.19	233.15	connection profileq1
	pgmacct2	prod.acct.pgm	verdi	0.12	6.12	connection profiletq2
	pgmacct3	prod.acct.pgm	diana	0.05	170.45	connection profiletq3
	pgmacct4	prod.acct.pgm	diana	0.34	145.23	connection profileq4

ctmruninf

The ctmruninf utility displays runtime data from the Statistical Details table of the Control-M/Server database. An option is available to delete data from this table. The jobs scanned for both options can be limited to a range of dates as described below. To run the ctmruninf utiilty. see: Running the ctmruninf utility (on page 605).

Statistical data is only accumulated when the Control-M/Server system parameter Statistics is set to **Y**.

For more information about runtime statistical data, see the information about runtime statistics in Control-M Forecast parameters.

Running the ctmruninf utility

This procedure describes how to run the ctmruninf utility, which enables you to display runtime data from the Statistical Details table of the Control-M/Server database.

➢ To run the the ctmruninf utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the one of the following commands:
   - `ctmruninf -list <fromDate> <toDate> [<filter>] [-total]`
   - `ctmruninf -list "*" [<filter>] [-total]`
   - `ctmruninf -delete <fromDate> <toDate>`
   - `ctmruninf -PURGE`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more information on the ctmrunif utility, see ctmruninf utility parameters (on page 606) and ctmrunif utility example (on page 608).

ctmruninf utility parameters

The following table describes the ctmruninf utility parameters:

Parameter	Description
-list	Displays data from the Statistical Details table within the dates specified in the From Date and To Date parameters. The data listed can be limited by using the Filter subparameter (described in this table).
-delete	Deletes data from the Statistical Details table in the range specified in the From Date and To Date parameters.
<From Date>	Start date of statistical data to be displayed or deleted. Format: yyyymmddhhmmss
<To Date>	End date of statistical data to be displayed or deleted. Format: yyyymmddhhmmss
"*"	Asterisk enclosed in quotation marks. Specifies that the utility should list all statistical data currently available, without regard to date.
<Filter>	Specify one of the following options and its associated subparameter or leave blank to display the statistics for all jobs in the range. ▪ -JOBNAME <jobName> Identify the job by the first 10 characters in its Job Name parameter. ▪ -MEMNAME <memname> Identify the job by its Mem Name parameter. ▪ -MEMLIB <memlib> Identify jobs by their Mem Lib parameter specifications. ▪ -HOSTID <hostid> Identify jobs by their Host ID parameter (agent computer). ▪ -ORDERID <orderid> Identify jobs by their Order ID parameter. Each of the subparameters in the filter can include the following wildcard characters: ▪ * represents any number of characters (including none). Any parameter including * should be enclosed in quotation marks (see the third example below). ▪ ? represents any single character

Parameter	Description
-total	Displays the total CPU and elapsed times for the jobs selected.
-PURGE	Purge data from the Statistical Details table based on the number of job executions. Running the ctmruninf utility with the -PURGE option performs the statistics cleanup as if it was done during New Day with the RUNINF_PURGE_MODE set to 0 (default). The Statistics algorithm (JOBNAME or MEMNAME) and the RUNINF_PURGE_LIMIT parameter are taken from the **config.dat** table, if configured. ▪ You can speed up the New Day procedure by specifying N for the STATISTICS_CLEANUP_IN_NEWDAY parameter and running ctmruninf -PURGE in a job that is run daily. ▪ Only the last *n* run information records of a job are kept, where *n* is the value of RUNINF_PURGE_LIMIT (default 20).

ctmrunif utility example

The following example describes command displays runtime data for the period January 21, 2008 through January 25, 2008 (assuming that this data is available):

```
ctmruninf -list 20080121000000 20080125235959
```

The following command deletes the statistical data for January 31, 2008:

```
ctmruninf -delete 20080131000000 20080131235959
```

The following command causes the utility to display and total runtime data for all jobs on agent computer diana.

```
ctmruninf -list "*" -HOSTID "diana" -total
```

A report similar to the following is displayed:

```
             TIMESTAMP     JOBNAME    ORDERID  RUN# HOSTID       MEMNAME
MEMLIB         CPU   ELAPSED

 ------------- ---------- -------- ---- ------------ ----------
-------------- ----- -------

2000012160524 acct12     00000007    1 diana        pgmacct
prod.acct.pgm   0.19  233.15

2000012161205 gen786     0000000b    1 diana        genx       prod.general
0.12    6.12

 2000012162311 acct14     00000011    1 diana        pgmacct
prod.acct.pgm   0.05  170.45

2000012164512 acct15     00000012    1 diana        pgmacct
prod.acct.pgm   0.14  145.23

 ------------- ---------- -------- ---- ------------ ----------
-------------- ----- -------

Total records printed :
4                                                      0.50  555.35
```

ctmstats

The ctmstats utility displays and deletes statistical data from the Statistical Summary table of the Control-M/Server database. The data scanned for both options can be limited to a range of dates. The Statistical Summary table is created using the ctmjsa utility. To run the ctmstats utility see: Running the ctmstats utility (on page 609).

Statistical data is only accumulated when the Control-M system parameter Statistics is set to **Y**. For more information, see System Parameters referred to in ctmsys (on page 445).

Running the ctmstats utility

This procedure describes the ctmstats utility, which displays and deletes statistical data from the Statistical Summary table of the Control-M/Server database.

➢ To run the ctmstats utility:

1. Do one of the following:
   - **a.** Log on to a Control-M for Databases account (*UNIX*)
   - **a.** Open a command prompt window (*Windows*) where Control-M/EM is installed.
   - **b.** For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.
2. Enter the one of the following commands:
   - `ctmstats -list <fromDate> <toDate> [<filter>] [-total]`
   - `ctmstats -list "*" [<filter>] [-total]`
   - `ctmstats -delete <fromDate> <toDate>`

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the ctmstats utility, see ctmstats utility parameters (on page 610) and ctmstats utility example (on page 611).

ctmstats utility parameters

The following table describes the ctmstats utility parameters:

Parameter	Description
-list	Displays data from the Statistical Summary table within the dates specified by the From Date and To Date parameters. The data listed can be limited with use of the Filter sub-parameter (see below).
-delete	Deletes data from the Statistical Summary table in the range specified by the From Date and To Date parameters.
<*From Date*>	Starting date of statistical data to be displayed/deleted. The date is specified in yyyymmddhhmmss format.
<*To Date*>	Ending date of statistical data to be displayed/deleted. The date is specified in yyyymmddhhmmss format.
"*"	Asterisk enclosed in quotation marks. Specifies that the utility should list all statistical data currently available, without regard to date.
<Filter>	Specify one of the following options and its associated subparameter or leave blank to display the statistics for all jobs in the range.
	-JOBNAME <jobName> Identify the job by its Job Name parameter.
	-MEMNAME <memname> Identify the job by its Mem Name parameter.
	-MEMLIB <memlib> Identify jobs by their Mem Lib parameter.
	-HOSTID <hostid> Identify jobs by their host id parameter (agent computer).
	Each of the subparameters in the filter can include the following wildcard characters: ▪ * – Represents any number of characters (including no characters). Any parameter including * should be enclosed in quotation marks (see example below). ▪ ? – Represents any single character
-total	Displays a line that contains the total CPU and elapsed times for the jobs selected.

ctmstats utility example

The following example describes the command displays statistical data for the period January 21, 2008 through January 25, 2008 (assuming that this data is available):

```
ctmstats -list 20080121000000 20080125235959
```

A report similar to the following is displayed:

```
TIMESTAMP       JOBNAME  HOSTID  MEMNAME  MEMLIB         AVG CPU  AVG
ELAPSED
--------------  ------   ------  -------  -------------  -------  --------
---
20080122141214  acct12   diana   pgmacct  prod.acct.pgm  0.19       233.15
20080122032025  gen786   diana   genx     prod.general   0.12         6.12
20080121123111  acct14   diana   pgmacct  prod.acct.pgm  0.05       170.45
20080121113512  acct15   diana   pgmacct  prod.acct.pgm  0.14       145.23
```

The following command displays statistical data for all jobs on agent computer diana:

```
ctmstats -list "*" -HOSTID diana -total
```

The following command deletes the statistical data for January 31, 2008:

```
ctmstats -delete 20080131000000 20080131235959
```

Upgrade

The migrate_dc utility promotes Control-M/Server job processing definition formats from earlier versions of Control-M.

The migrate_dc utility is part of the Control-M upgrade process. The migrate_dc utility converts Control-M/Server job processing definition formats within the Control-M/EM database. The Control-M/Server job processing definition formats are converted from the Control-M/Server data formats in the earlier version to the data formats in the new version. For more information about upgrade, see the *Control-M Upgrade*.

Do not use this utility if you have not yet installed and upgraded to a new version of Control-M/EM and Control-M/Server.

The upgrade process was previously called Migration. Control-M/Server is sometimes called Data Center.

migrate_dc

Use the migrate_dc utility to promote the job processing definition formats in Control-M/EM from an earlier versions to Control-M/Server to the format of the current version. To promote using Control-M Configuration Manager, see Promoting Control-M/Server data formats on Control-M/EM. To run the utility, see Running the migrate_dc utility (on page 612).

There is no rollback procedure from changes made by the migrate_dc utility. Before using the migrate_dc utility, backup all your data

The destination Control-M/Server must be defined in COMM folder before executing the migrate_dc utility.

By default, all job definitions are converted to a format that is consistent with the standards for the current version.

If you upgraded Control-M/Server on UNIX or on Windows to a version earlier than 6.4.01, or if you upgraded Control-M for OS/390 to a version earlier than 6.4.01, and then want to run the migrate_dc utility in Control-M for Databases for the upgraded Control-M definitions, you must specify **-version {630|640|700|800}** in the migrate_dc command line to prevent the current version format from being applied to the jobs for this Control-M.

Running the migrate_dc utility

This procedure describes how to run the migrate_dc utility, which enables you to promote the job processing definition formats in Control-M/EM from an earlier versions to Control-M/Server to the format of the current version.

➢ To run the migrate_dc utility:

1. Do one of the following:
   a. Log on to a Control-M/EM account (*UNIX*).
   a. Open a command prompt window (*Windows*) where Control-M/EM is installed.

   b. For Windows client installations, open a command prompt window and navigate to the **<***EM Home***>\Default\bin** directory.

2. Enter either of the following commands:

- To promote Control-M/Server job processing definition formats from all database folders:

  ```
  migrate_dc -u <dboName> -p <dboPassword> -dc <dataCenter>
  [-hostname <dataCenterHostName>]
  [-port <dataCenterPortNumber>]
  [-version {630|640|700|800}]
  ```

- To promote Control-M/Server job processing definition formats from folder-to-folder:

  ```
  migrate_dc -u <dboName> -p <dboPassword>
  [-version {630|640|700|800}]
  [-old_dc <name1>]
  [-new_dc <name2>]
  [-folder <folderName>]
  [-lib <libraryName>]
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the migrate_dc utility, see migrate_dc utility parameters (on page 614).

migrate_dc utility parameters

The following table describes the parameters of the migrate_dc utility:

Parameter	**Description**
hostname	the Control-M/Server host name (maximum length: 255 characters)
port	the Control-M/Server port number (range: 1024 – 65533)
version	the newly upgraded Control-M/Server version number This indicates to Control-M/EM that Control-M/Server was upgraded and accordingly adjusts it to the new Control-M/Server version.
old_dc	the source Control-M/Server (data center) name, as defined in the COMM folder
new_dc	the destination Control-M/Server name If the Control-M/Server <*name2*> is not defined in COMM folder, the following error message is displayed: The target Control-M/Server is not defined
folder	the upgraded folder name
lib	optional parameter for specifying or identifying an z/OS-folder

migrate_dc example

The following is an example of migrate_dc usage:

Use the migrate_dc utility to promote Control-M/Server job processing definition formats in the Control-M/EM database on computer saturn from version 6.3.01 Control-M/Server data formats to version 8.0.00 Control-M/Server data formats.

The following table lists the names and definitions used in this example:

Parameter	Name
dataCenter (Control-M/Server)	jupiter
dboName	asteroid
dboPassword	star2

Parameter	Name
dataCenterHostName	saturn
dataCenterPortNumber	7530

Promoting from earlier versions to version 8.0.00

This procedure describes how to promote from earlier versions to version 8.0.00 using the migrate_dc utility.

➢ To promote from earlier versions using the migrate_dc utility:

1. Log on to computer saturn where version 8.0.00 of Control-M/EM or where Control-M Configuration Manager is installed.
2. If Control-M/Server jupiter's host and port have changed during the Control-M/Server upgrade and the correct host and port are not specified in the **-hostname** and **-port** options of migrate_dc, select the relevant Control-M/Server in the Control-M Configuration Manager GUI and update the host and port definitions.
3. Verify that the version 6.3.01 Control-M/Server being upgraded is disconnected from Control-M/EM and that the corresponding gateway is down.
4. In Control-M Configuration Manager, if the Control-M/Server version 8.0.00 on either a UNIX computer or a Windows computer, or version 62A or later (Control-M for OS/390), is in *Managed* mode, change the status to *Unmanaged*.
5. Open a command prompt window on Windows, or log on to the account where Control-M for Databases is installed on UNIX.
6. To upgrade the data formats for Control-M/Server jupiter, run one of the following commands:
    - to promote from all version 6.3.01 database folders to version 8.0.00

      ```
      migrate_dc -u asteroid -p star2 -dc jupiter
      -hostname saturn
      -port 7530
      -version 800
      ```

    - to promote Control-M/Server job processing definitions from folder-to-folder

      ```
      migrate_dc -u asteroid -p star2
      -version 800
      -old_dc jupiter1
      -new_dc jupiter2
      -folder starFolder
      -lib starLibrary
      ```

Chapter 13

XML file preparation

XML files have the following characteristics:

- XML is a structured format for organizing and specifying data.
- Data in an XML file is classified by type.
- Words enclosed in angle brackets (< >), called tags, are used to classify and organize the data.

In the XML files used by the Control-M/EM utilities, tags are used to classify job processing definition, Calendar, folder, and SMART Folder parameters, and their values.

- You do not need to know XML to use these utilities. The instructions in this chapter provide you with the information that you need to know to produce all utility files.
- Some of the parameter names changed for Control-M version 8.0.00, terminology from previous versions is still supported. For a complete list of the parameter names, see Abbreviations and conventions (on page 10).

Control-M/EM utility commands

Each utility is composed of a combination of at least two of the following parts:

- An invocation command
- An input file containing either data to enter into the Control-M/EM database or arguments for selecting specific data from the database
- An output file containing data specified in the arguments file, if it was used
- Optional switches for controlling how the utility runs

For example, the defjob utility has three parts; the invocation command, a file of job processing definitions that are imported into the Control-M/EM database, and an optional switch. You prepare the file containing the job processing definitions.

The exportdefjob utility uses an invocation command, a file containing arguments for specifying the job processing definitions that are exported from the Control-M/EM database, an optional switch, and an output file containing the exported job processing definitions. You prepare the arguments file. The output file is created by the exportdefjob utility.

Preparing an input file

Control-M/EM utilities read input text files that are used to enter information into the Control-M/EM database. Control-M/EM export utilities export data from the database in text files. Both the input and the output files are formatted with XML.

For example, the defcal input file specifies new Calendar definitions to enter into the database.

The indentations used to format the input file help you understand the hierarchical relationships between elements in the file. These indentations are not mandatory and do not affect how the file is processed.

File structure

The different parts of the input file are defined by tags composed of punctuation marks. The TERMS input file displayed in the example selects all non-cyclic jobs with job name Job5. The action performed on the selected jobs is determined by the type of utility that is calling the TERMS file. Using this TERMS file with deldefjob deletes all job processing definitions in the database for non-cyclic jobs with job name Job5.

This file contains one TERMS statement. The statement specifies that non-cyclic jobs with Job Name Job5 are to be selected.

```
<TERMS>
<TERM>
<PARAM NAME="JOBNAME" OP="EQ" VALUE="Job5"/>
<PARAM NAME="CYCLIC" OP="EQ" VALUE="0"/>
</TERM>
</TERMS>
```

The statement begins with the word TERMS enclosed in angle brackets (<TERMS>). The end of the file is indicated by the closing TERMS tag. That this is the end of the TERMS statement is indicated by the presence of the slash (/), so that the closing statement looks like </TERMS>.

Between the <TERMS> tags is a search term for identifying and selecting specific job processing definitions. It is indicated by the tags <TERM></TERM>. Between the TERM tags are the parameters of the search, indicated by the <PARAM/> tag. All the attributes of the tag are contained within the single set of brackets. No closing tag is needed. As a result, the slash (/) is included in the single tag, preceding the closing angle bracket, <PARAM/>.

As noted, the PARAM tag contains the search terms, NAME, OP, and VALUE.

- NAME is the name of a job processing definition parameter.
- OP is an operator. The most common operators are described in the following table.
- VALUE is the value of the parameter to which a comparison is being made.

NAME="JOBNAME" OP="EQ" VALUE="Job5" searches for job processing definitions that have the Job Name, Job5.

Utility operators

Operator	Description
EQ	Equals. Select cases that include the specified value.
NEQ	Not equal. Select cases that include any value different from the one specified.
LIKE	Similar. Select cases that have an attribute common to the one specified. You must use a wildcard, such as * in the value that you specify. `JOBNAME LIKE="JOB1*"` selects all jobs with a job name that begins with JOB1. JOB13 would be selected, but not JOB25.

What to include in the file

Each utility is described in this book with a table of elements (job, calendar, and folder parameters) and attributes (subparameters). Use the valid values described in the tables, making sure to use the same case and spelling.

If the valid value is a string, see the description of the appropriate parameter in *Control-M Parameters* for information about valid values and their formats.

When working in an I18N environment, the following header must be placed at the top of the argument file:

```
<?xml version='1.0' encoding='UTF-8'?>
```

For example:

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE TERMS SYSTEM "terms.dtd">

<TERMS>

<TERM>
```

```
<PARAM NAME="JOBNAME" OP="EQ" VALUE="Job5"/>
<PARAM NAME="CYCLIC" OP="EQ" VALUE="0"/>
</TERM>
</TERMS>
```

Validating your file

The contents of an XML file are determined by a set of criteria that are contained in a document definition type file. This file has a **.dtd** extension.

The **.dtd** file is used by Control-M/EM to validate the input file (or arguments file) when the utility runs.

The **.dtd** file includes the following information:

- Names of all of the elements and attributes that can be entered in an input or arguments file in XML format.
- Valid values for an element or attribute.
- Whether the valid values for an element or attribute are mandatory or optional.
- Hierarchical relationship between the various elements and attributes in the file.

Formatting and value information for Control-M parameters is described in *Control-M Parameters.*

Control-M/EM utility .dtd files

Each utility input file has its own **.dtd** file. The TERMS arguments files share the same **.dtd** file. Utility **.dtd** files are stored in the *Control-M|emHome***\emccmcli\Resource** directory (on Windows).

File name	Description
copycal.dtd	Validates the copydefcal input file.
copyjob.dtd	Validates the copydefjob input file.
defcal.dtd	Validates the defcal input file.
defjob.dtd	Validates the defjob input file.
deffolder.dtd	Validates the deffolder input file.
duplicatejob.dtd	Validates the duplicatedefjob input file.
terms.dtd	Validates exportdefjob, exportdefcal, and exportdeffolder argument files.
update.dtd	Validates the updatedef input file.

➢ To create a file:

1. Open any text editor or an XML editor.
2. Enter the data for the utility that you are using, the format described in the parameter description tables, and examples that are provided with the utility.

3. Check the syntax of the file for errors. If errors remain, they are identified when the file is submitted to the utility. Possible errors include:

- Misplaced or missing tag.
- Misplaced or missing part of a tag (for example, a missing slash /).
- Parameter value that is not specified as a valid value for that parameter (for example, the letter Y instead of a 1.

Control-M for Databases validates every file you submit, rejecting those that have errors. When a file is rejected, the lines containing errors are specified for you.

4. Save the file.

The file must be saved as a text file. It can have any file extension you want. However, BMC recommends that you use **.xml**

Reserved characters

Certain characters are reserved for formatting the XML file. These characters cannot be used in job parameter values submitted with the XML-based utilities.

Instead, each reserved character must be replaced by a code. The reserved characters and the codes are replaced and are listed in the following table:

Character	Replacement code
" (double-quote)	&quot;
’ (single-quote, apostrophe)	&apos;
< (left-angle bracket)	&lt;
> (right-angle bracket)	&gt;
& (ampersand)	&amp; The ampersand character can be used in the character replacement codes.

Using reserved character codes in an XML file.

Incorrect:

```
INCOND NAME="if5<6’run’" ODATE="ODAT" AND_OR="AND" OP="("/
```

Correct:

```
INCOND NAME="if5&lt;6&apos;run&apos;" ODATE="ODAT" AND_OR="AND"
OP="("/
```

Wildcards

Multiple jobs can be selected and copied using the * (asterisk) wildcard character to represent multiple values. The asterisk is used to represent zero or more alphanumeric characters.

The asterisk in search criteria

An asterisk can be used to replace characters in the middle of an expression.

Only one asterisk can be used in an expression.

The job name of a specific job definition is AAABBB. If you include the any of the following arguments in an updatedef utility argument file, you select job AAABBB:

```
<JOB_NAME FROM="AAABBB"/>

<JOB_NAME FROM="*BBB"/>

<JOB_NAME FROM="AAA*"/>
```

There are three job processing definitions. Their Job Names are:

```
AAABBB, AAACCC, and BBBCCC
```

The following argument selects jobs AAACCC and BBBCCC, and selects any other jobs with a Job Name that ends with the letters CCC.

```
<JOB_NAME FROM="*CCC"/>
```

The asterisk in find/replace operations

The asterisk has a special function when used in "find and replace" operations in selected utilities. The following utilities use find and replace operations:

- copydefcal
- copydefjob
- duplicatedefjob
- updatedef

In the FROM (find) statement of an argument, an asterisk replaces a text string (as shown in the example above). The asterisk in the TO statement of the argument represents the same string as the asterisk in the FROM statement of the argument.
The placement of the asterisk can be changed.

There are three job processing definitions. Their Job Names are:

```
AAABBB, AAACCC, BBBCCC
```

Modify the job names of some of these jobs using the following argument:

```
<JOB_NAME FROM="*CCC" TO="*DDD"/>
```

- Job AAACCC becomes Job AAADDD
- Job BBBCCC becomes Job BBBDDD
- Job AAABBB is not modified.

There are three job processing definitions. Their Job Names are:

```
AAABBB, DDDCCC, BBBCCC
```

```
<JOB_NAME FROM="*CCC" TO="DDD*"/>
```

- Job DDDCCC becomes Job DDDDDD
- Job BBBCCC becomes Job DDDBBB
- Job AAABBB is not modified.

Forecast

The forecastcli utility enables you to perform a Load Forecast operation in batch mode. The main outputs are, for the specified date:

- Service information (only available if you have the BMC Control-M Batch Impact Manager Add-on installed)

  Service information includes a summary of the business services identified by Control-M/Forecast. If at least one service is identified as late, a warning is indicated in the output.

- Job information (always available)

  Job information is ordered for a specific future date. You can specify whether or not the estimated run times are included in the output..

To run the forecastcli utility, see Running the forecastcli utility (on page 623)

Running the forecastcli utility

This procedure describes how to run the forecastcli utility, which enables you to perform a Load Forecast operation in batch mode.

Before you begin

- To run the forecastcli utility, the Control-M/Forecast server must be up and running.

➢ To run a forecast as a batch file:

- Type the following command;

  forecastcli -u <*user*> [[-p <*password*>] | -pf <*password_file*>] -s <*server_name*> -odate [YYYYMMDD|+n] [Filter] [-scenario <*name*>] [-run_time [Min|Avg|Max]] -job_info_file <*file_name*> [-service_info_file <*file_name*>] [/hide_times]

  For UNIX, add em and a space before specifying forecastcli. For example:

  ```
  em forecastcli -u emuser -p empass -s emserv1 -odate 20090217 -job_info_file
  out.csv
  ```

For the Workload Automation parameter name, see Parameter name cross references (on page 34). For more details on the forecastcli utility, see the following:

- forecastcli parameters (on page 624)
- forecastcli filter options (on page 625)
- forecastcli examples (on page 625)

forecastcli parameters

The following table describes the parameters of the forecastcli commands:

Parameter	Description	
-u *<user>*	Indicates the username of the Control-M/Enterprise Manager client. **Required**.	
-p *<password>*	Indicates the password of the Control-M/Enterprise Manager client.	
-pf *<password_file>*	Indicates the password file name used instead of password.	
-s *<server_name>*	Indicates the name of the Control-M/Enterprise Manager server. **Required**.	
-odate	Indicates the forecast report date (or number of days from current date). **Required**. Valid values are:	
	YYYYMMDD	A specific working day in YYYYMMDD format.
	+n	Number of days from current date.
-scenario *<name>*	Indicates the name of the forecast scenario that is applied to the forecast.	
-run_time	Indicates type of run time that is used for the forecast. Valid values are:	
	Avg	Average (Default)
	Min	Minimum
	Max	Maximum
-job_info_file *<file_name>*	Indicates the file name where the job information output is saved. The output is in CSV format. **Required**.	
-service_info_file *<file_name>*	Indicates the file name where the service information output is saved. The output is in CSV format.	
-hide_times	The estimated job start and end time will not be printed in the job information output file.	
/? or /h	Displays usage.	

forecastcli filter options

The following table describes the forecastcli Filter options.

Parameter	Filter results with
-ctm_name *<name>*	Control-M name
-folder_name *<name>*	Folder name
-appl_name *<name>*	Application name
-subappl_name *<name>*	Sub application name
-mem_lib *<name>*	mem lib
-**mem_name** *<name>*	mem name
-job_name *<name>*	job name

forecastcli examples

The following are forecastcli examples:

To know, on a daily basis, which services or jobs will be ordered on the following day, run the following command. The output files, services.csv and jobs.csv, list the services and jobs, indicating their run times and statuses.

```
forecastcli -u emuser -p empass -s emserv1 -odate +1 -job_info_file
jobs.csv -service_info_file services.csv
```

The following command generates a forecast for February 17, 2009. The job information is written to the out.csv file.

```
forecastcli -u emuser -p empass -s emserv1 -odate 20090217
-job_info_file out.csv
```

The following command generates a forecast to obtain a list of the jobs that will run on all data centers with names beginning with "A" for February 17, 2009. The run times are not included in the output file in order to compare it to another job list forecasted for another date.

```
forecastcli -u emuser -p empass -s emserv1 -odate 20090217 -dc_name
A* -job_info_file out.csv /hide_times
```

Chapter 15

BMC Control-M Batch Discovery

Using BMC Control-M Batch Discovery, critical batch services defined and monitored with Control-M/EM and BMC Control-M Batch Impact Manager can be output in CSV files and imported into the BMC Atrium CMDB to enhance control over change processes and system failures, and provide a view of the critical batch processes in the IT environment. To run the BMC Control-M Batch Discovery utility, see Running the Batch Discovery utility (on page 626).

Running the Batch Discovery utility

This procedure describes how to run the BMC Control-M Batch Discovery utility.

➢ To invoke BMC Batch Discovery:

1. Begin according to your operating system:
   - **Windows**: Open the Command Prompt window and navigate to the **bin** folder, located in the Control-M/EM installation folder where BMC Control-M Batch Discovery was installed.
   - **UNIX**: Log on to a Control-M/EM administrator account.
2. Enter one of the following commands:
   - **Windows**:

     ```
     em_batchdiscovery -u <emUser> {-p <emPassword> -gsr
     <guiServerName>[-cms <cms>][-pf <passwordFile>]
     ```

   - **UNIX**:

     ```
     em batchdiscovery -u <emUser> {-p <emPassword> -gsr <guiServerName>
     [-cms <cms>] [-pf <passwordFile>]
     ```

   For more information on the parameters, see BMC Batch Discovery parameters (on page 627).
3. Press **Enter**.

   The "BMC Batch Discovery ended successfully" message indicates that BMC Control-M Batch Discovery completed the creation of csv files containing information from the Control M/EM database.

   The CSV files can be found in Control-M/EM home directory.

   If BMC Batch Discovery does not end successfully, a return code is displayed, as described in BMC Batch Discovery return codes (on page 627). The code can be used to detect how the application failed.

BMC Batch Discovery parameters

The following table describes the BMC Control-M Batch Discovery parameters:

Parameter	Description
-u	Control-M/EM administrator name.
-p	Control-M/EM administrator password.
-gsr	Name of Control-M/EM GUI Server.
-cms	Optional. Name of Control-M Configuration Manager. Optional. (Default: CMS)
-pf	Optional. Name of flat file containing a list of unencrypted passwords, on separate lines, in the following format: *EM password* *AR System password* **Note:** To use the password file, specify the **-pf** parameter instead of the **-p.** If only **-u** is specified, online prompts are issued for the passwords. If the **-pf** parameter is specified with **-p,** the **-p** parameter is ignored.

BMC Batch Discovery return codes

The following table describes the BMC Control-M Batch Discovery return codes:

Return code	Description
0	Success
1	Failure
2	Operating system not supported
255	Failure

Chapter 16

Unsupported utilities

The following utilities are provided "as is."

BMC Software does not support these utilities and assumes no responsibility for problems that may occur as a result from using these utilities. BMC Software advises users not to use these utilities:

```
addevice
addto_interfaces_file
ags
ajf
ctm_backup_aut
ctm_grj
ctm_jcl
ctm_mirrordb_bck
ctm_newday
ctm_shout
ctm_output_down
ctm_output_hndl
ctm2snmp
ctmdbcount
ctmeditjcl
ctmgtsch
ctmjckdl
ctmjcopy
ctmlbsel
ctmmksch
ctmshdst
ctmsnmp
ctmtunnelreq
ctmweb
CtoSrvDa
dbbackupora
dbbackupsyb
```

```
dbcheckora
dbchecksyb
dbloadora
dbloadsyb
dbrestoreora
dbrestoresyb
dbunloadora
dbunloadsyb
dbupstatora
dbupstatsyb
dsdisp
dsmk
ecacontb
shdest
upg_ping
```